H3Cアクセスコントローラ

セキュリティコンフィギュレーションガイド

New h3c Technologies Co., Ltd.
http://www.h3c.com

ドキュメントバージョン:6W103-20200507製品バージョン:R5426P02

内容

SSLの構成

SSLについて

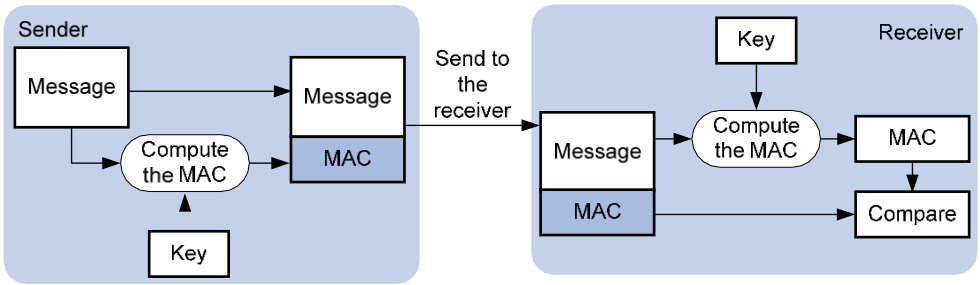
SSL(Secure Sockets Layer)は、HTTPなどのTCPベースのアプリケーション層プロトコルに通信セキュリティを提供する暗号化プロトコルです。SSLは、インターネットを介した安全なデータ伝送を提供するために、e-businessやオンラインバンキングなどのアプリケーションで広く使用されています。

SSLセキュリティサービス

SSLは次のセキュリティサービスを提供します。

- **Privacy**: SSLでは、対称暗号化アルゴリズムを使用してデータが暗号化されます。対称暗号化アルゴリズムで使用されるキーの暗号化には、RSAの非対称キーアルゴリズムが使用されます。RSAの詳細は、「公開鍵の管理」を参照してください。
- **Authentication**: SSLでは、証明書ベースのデジタル署名を使用してSSLサーバーおよびクライアントが認証されます。SSLサーバーおよびクライアントは、PKIを介してデジタル証明書を取得します。PKIおよびデジタル証明書の詳細は、「PKIの構成」を参照してください。
- **Integrity**: SSLでは、メッセージの整合性を検証するためにメッセージ認証コード(MAC)が使用されます。MACアルゴリズムおよびキーを使用して、任意の長さのメッセージが固定長メッセージに変換されます。元のメッセージを変更すると、計算された固定長メッセージも変更されます。
  図1に示すように、メッセージ整合性検証プロセスは次のようになります。
  a. 送信側は、MACアルゴリズムとキーを使用してメッセージのMAC値を計算し、メッセージにMAC値を付加して受信側に送信します。
  b. 受信側は、同じキーとMACアルゴリズムを使用して、受信メッセージのMAC値を計算し、メッセージに付加されたMAC値と比較します。
  c. 2つのMAC値が一致する場合、受信者はメッセージをそのまま見なします。一致しない場合、受信者はメッセージが改ざんされたと見なし、メッセージを廃棄します。

図1 MACアルゴリズムの図

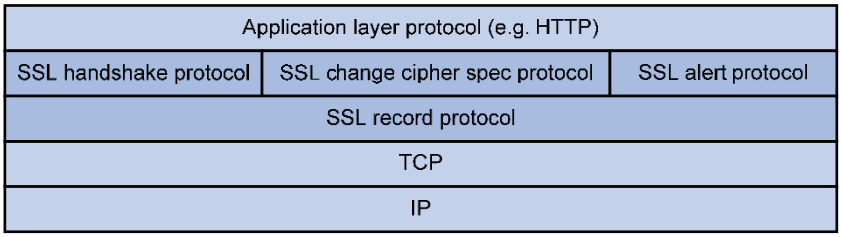


SSLプロトコルスタック

SSLプロトコルスタックには、次のプロトコルが含まれます。

- 下位層のSSLレコードプロトコル。
- SSLハンドシェイクプロトコル、SSL暗号仕様変更プロトコル、および上位層のSSLアラートプロトコル。

図2 SSLプロトコルスタック



次に、SSLプロトコルの主な機能について説明します。

- **SSLレコードプロトコル**: 上位層から受信したデータをフラグメント化し、MACを計算してデータに追加し、データを暗号化します。
- **SSLハンドシェイクプロトコル**: 安全な通信に使用される暗号スイートをネゴシエートし、サーバーおよびクライアントを認証し、サーバーとクライアント間でキーを安全に交換します。ネゴシエートする必要がある暗号スイートには、対称暗号化アルゴリズム、キー交換アルゴリズムおよびMACアルゴリズムが含まれます。
- **SSL暗号仕様プロトコルの変更**: ネゴシエートされた暗号スイートおよびキーに基づいて後続のパケットが保護されることを受信者に通知します。
- **SSLアラートプロトコル**: 受信側にアラートメッセージを送信します。アラートメッセージには、アラートの重大度レベルと説明が含まれます。

SSLプロトコルのバージョン

SSLプロトコルバージョンには、SSL 2.0、SSL 3.0、TLS 1.0(またはSSL 3.1)、TLS 1.1およびTLS 1.2が含まれます。SSL 3.0は安全でないことがわかっているため、SSLサーバーのSSL 3.0を使用不可にしてセキュリティを確保できます。

制約事項および注意事項:SSL構成

デフォルトでは、SSLサーバーはすべてのSSLプロトコルバージョンを実行しているクライアントと通信できます。サーバーがクライアントからSSL 2.0 Client Helloメッセージを受信すると、クライアントに対して、通信に後のバージョンを使用するように通知します。

SSLタスクの概要

SSLサーバーの構成

- SSLサーバーポリシーの構成
- (任意)SSLサーバーのSSLプロトコルバージョンをディセーブルにします。
- (任意)SSLセッション再ネゴシエーションのディセーブル化

SSLクライアントの構成

SSLクライアントポリシーの構成

SSLサーバーポリシーの構成

このタスクについて

SSLサーバーポリシーは、デバイスがSSLサーバーとして動作するときにデバイスによって使用されるSSLパラメータのセットです。SSLサーバーポリシーは、HTTPSなどのアプリケーションに関連付けられた後にのみ有効になります。

手順

1. システムビューに入ります。

   system-view

2. SSLサーバーポリシーを作成し、そのビューを入力します。

   ssl server-policy policy-name

3. SSLサーバーポリシーのPKIドメインを指定します。

   pki-domain domain-name

   デフォルトでは、SSLサーバーポリシーにPKIドメインは指定されていません。

   SSLサーバー認証が必要な場合は、PKIドメインを指定して、ドメイン内のSSLサーバーのローカル証明書を要求する必要があります。

   PKIドメインの設定方法については、「PKIの構成」を参照してください。

4. SSLサーバーポリシーがサポートする暗号スイートを指定します。

   ciphersuite { dhe_rsa_aes_128_cbc_sha | dhe_rsa_aes_128_cbc_sha256 | dhe_rsa_aes_256_cbc_sha | dhe_rsa_aes_256_cbc_sha256 | ecdhe_ecdsa_aes_128_cbc_sha256 | ecdhe_ecdsa_aes_128_gcm_sha256 | ecdhe_ecdsa_aes_256_cbc_sha384 | ecdhe_ecdsa_aes_256_gcm_sha384 | ecdhe_rsa_aes_128_cbc_sha256 | ecdhe_rsa_aes_128_gcm_sha256 | ecdhe_rsa_aes_256_cbc_sha384 | ecdhe_rsa_aes_256_gcm_sha384 | exp_rsa_des_cbc_sha | exp_rsa_rc2_md5 | exp_rsa_rc4_md5 | rsa_3des_ede_cbc_sha | rsa_aes_128_cbc_sha | rsa_aes_128_cbc_sha256 | rsa_aes_256_cbc_sha | rsa_aes_256_cbc_sha256 | rsa_des_cbc_sha | rsa_rc4_128_md5 | rsa_rc4_128_sha } *

   デフォルトでは、SSLサーバーポリシーはすべての暗号スイートをサポートします。

5. (任意)SSLサーバーがキャッシュできるセッションの最大数とセッションキャッシュタイムアウト時間を設定します。

   session { cachesize *size* **| timeout** *time* **} ***

   デフォルトでは、SSLサーバーは最大500セッションをキャッシュできます。セッションキャッシュのタイムアウト時間は3600秒です。

6. 必須またはオプションのSSLクライアント認証をイネーブルにします。

   client-verify { enable | optional }

   デフォルトでは、SSLクライアント認証は使用不可です。SSLサーバーは、SSLクライアントに対して電子証明書ベースの認証を実行しません。

   電子証明書を使用してクライアントを認証する場合、SSLサーバーはクライアントによって提示された証明書チェーンを検証します。また、証明書チェーン内の証明書(ルートCA証明書を除く)が失効していないことも検証します。

7. (任意)SSLネゴシエーション中にSSLサーバーが完全な証明書チェーンをクライアントに送信できるようにします。**中間証明書を使っての認証が成功しない場合には、この設定が有効な場合があります。**

   certificate-chain-sending enable

デフォルトでは、SSLサーバーはネゴシエーション中に完全な証明書チェーンではなくサーバー証明書をクライアントに送信します。

SSLクライアントポリシーの構成

このタスクについて

SSLクライアントポリシーは、デバイスがSSLクライアントとして動作するときにデバイスによって使用されるSSLパラメータのセットです。SSLクライアントはクライアントポリシーの設定を使用してサーバーへの接続を確立します。SSLクライアントポリシーは、FTPなどのアプリケーションに関連付けられた後にのみ有効になります。FTPの詳細は、「基本構成ガイド」を参照してください。

制限事項およびガイドライン

システムセキュリティを強化するためのベストプラクティスとして、SSLクライアントポリシーにSSL 3.0を指定しないでください。

手順

1. システムビューに入りに入ります。

   system-view

2. SSLクライアントポリシーを作成し、そのビューを入力します。

   ssl client-policy policy-name

3. SSLクライアントポリシーのPKIドメインを指定します。

   pki-domain domain-name

   デフォルトでは、SSLクライアントポリシーにPKIドメインは指定されていません。

   SSLクライアント認証が必要な場合は、PKIドメインを指定し、PKIドメイン内のSSLクライアントのローカル証明書を要求する必要があります。

   PKIドメインの設定方法については、「PKIの構成」を参照してください。

4. SSLクライアントポリシーの優先暗号スイートを指定します。

   prefer-cipher { dhe_rsa_aes_128_cbc_sha | dhe_rsa_aes_128_cbc_sha256 | dhe_rsa_aes_256_cbc_sha | dhe_rsa_aes_256_cbc_sha256 | ecdhe_ecdsa_aes_128_cbc_sha256 | ecdhe_ecdsa_aes_128_gcm_sha256 | ecdhe_ecdsa_aes_256_cbc_sha384 | ecdhe_ecdsa_aes_256_gcm_sha384 | ecdhe_rsa_aes_128_cbc_sha256 | ecdhe_rsa_aes_128_gcm_sha256 | ecdhe_rsa_aes_256_cbc_sha384 | ecdhe_rsa_aes_256_gcm_sha384 | exp_rsa_des_cbc_sha | exp_rsa_rc2_md5 | exp_rsa_rc4_md5 | rsa_3des_ede_cbc_sha | rsa_aes_128_cbc_sha | rsa_aes_128_cbc_sha256 | rsa_aes_256_cbc_sha | rsa_aes_256_cbc_sha256 | rsa_des_cbc_sha | rsa_rc4_128_md5 | rsa_rc4_128_sha } *

   デフォルトでは、SSLクライアントポリシーの優先暗号スイートはdhe_rsa_aes_128_cbc_sha、dhe_rsa_aes_256_cbc_sha、rsa_3 des_ede_cbc_sha、rsa_aes_128_cbc_sha、およびrsa_aes_256_cbc_shaです。

5. SSLクライアントポリシーのSSLプロトコルバージョンを指定します。

   Version { ssl3.0 | tls1.0 | tls1.1 | tls1.2 }

   デフォルトでは、SSLクライアントポリシーはtls1.0を使用します。

6. SSLクライアントがデジタル証明書を使用してサーバーを認証できるようにします。

   server-verify enable

   デフォルトでは、SSLサーバー認証はイネーブルです。

SSLサーバーのSSLプロトコルバージョンの無効化

このタスクについて

セキュリティを強化するために、SSLサーバーがセッションネゴシエーションに特定のSSLプロトコルバージョンを使用できないようにすることができます。

SSLサーバーのSSLプロトコルバージョンは、システムビューまたはSSLサーバーポリシービューで使用不可にできます。SSLサーバーは、SSLサーバーポリシーのSSLプロトコルバージョンのステータスが「使用可能」の場合にのみ、セッションネゴシエーションにSSLプロトコルバージョンを使用できます。SSLサーバーポリシーのSSLプロトコルバージョンのステータスは、次の順序で決定されます。

1. SSLサーバーポリシービューでの**version disable**コマンドの設定。
2. システムビューでの**ssl version disable**コマンドの設定。
3. 既定の設定(有効)。

SSLサーバーがセッションネゴシエーションに最低1つのSSLプロトコルバージョンを使用できることを確認してください。

制限事項およびガイドライン

SSLプロトコルバージョンを使用不可にしても、以前のSSLプロトコルバージョンの可用性には影響しません。たとえば、**ssl version tls1.1 disable**コマンドを実行すると、tls1.1は使用不可になりますが、tls1.0はSSLサーバーで引き続き使用可能です。

手順

1. システムビューに入ります。

   system-view
2. システムビューでSSLサーバーのSSLプロトコルバージョンを使用不可にします。

   ssl version { ssl3.0 | tls1.0 | tls1.1 | tls1.2 } * disable

   デフォルトでは、SSLサーバーはSSL 3.0、TLS 1.0、TLS 1.1、およびTLS 1.2をサポートします。
3. SSLサーバーポリシービューを入力します。

   ssl server-policy policy-name
4. SSLサーバーポリシーでSSLプロトコルバージョンを使用不可にします。

   version { ssl3.0 | tls1.0 | tls1.1 | tls1.2 } * disable

   デフォルトでは、SSLプロトコルバージョンは、**ssl version disable**コマンドを使用してシステムビューで明示的にディセーブルにしない限り、SSLサーバーポリシーでイネーブルになります。

SSLセッション再ネゴシエーションの無効化

このタスクについて

SSLセッション再ネゴシエーション機能を使用すると、SSLクライアントおよびサーバーは、以前にネゴシエートされたSSLセッションを簡略化されたハンドシェイクに再利用できます。

セッション再ネゴシエーションを無効にすると、システムの計算オーバーヘッドが増加しますが、潜在的なリスクを回避できます。

制限事項およびガイドライン

明示的に必要な場合のみ、SSLセッションの再ネゴシエーションを無効にします。

手順

1. システムビューに入ります。
   system-view
2. SSLセッションの再ネゴシエーションを無効にします。
   ssl renegotiation disable
   デフォルトでは、SSLセッションの再ネゴシエーションはイネーブルです。

SSL用の表示および保守コマンド

任意のビューで表示コマンドを実行します。

タスク	コマンド
SSLクライアントポリシー情報を表示します。	**display ssl client-policy [*policy-name*]**
SSLサーバーポリシー情報を表示します。	**display ssl server-policy [*policy-name*]**

PKIの構成

PKIについて

Public Key Infrastructure(PKI)は、ネットワークサービスを保護するためにデータを暗号化および復号化するための非対称キーインフラストラクチャです。

PKIでは、デジタル証明書を使用して公開鍵を配布および使用し、ユーザー認証、データ機密性およびデータ整合性などのセキュリティサービスとともにネットワーク通信およびEコマースを提供します。公開鍵の詳細は、「公開鍵の管理」を参照してください。

PKI用語

デジタル証明書

デジタル証明書は、CAによって署名された電子文書であり、公開鍵とその所有者のIDを結びつけます。

デジタル証明書には、次の情報が含まれます。

- 発行者名(証明書を発行したCAの名前)。
- サブジェクト名(証明書が発行される個人またはグループの名前)。
- サブジェクトのID情報。
- サブジェクトの公開鍵。
- CAの署名。
- 有効期間。

デジタル証明書は、ITU-T X.509の国際標準に準拠する必要があります。その中で、X.509v3が最も一般的に使用されています。

この章では、次のタイプの証明書について説明します。

- **CA証明書**: CAの証明書。PKIシステム内の複数のCAは、ルートCAを先頭にしてCAツリーを形成します。ルートCAは自己署名証明書を生成し、各下位レベルCAは、そのすぐ上にあるCAによって発行されたCA証明書を保持します。これらの証明書のチェーンは、信頼のチェーンを形成します。
- **Registration Authority(RA)証明書**: CAによってRAに対して発行された証明書。RAはCAのプロキシとして機能し、PKIシステムで登録要求を処理します。
- **ローカル証明書**: CAによってローカルPKIエンティティに対して発行されたデジタル証明書で、エンティティの公開鍵を含みます。
- **ピア証明書**: ピアのCA署名デジタル証明書。ピアの公開鍵が含まれています。

ルートCA証明書のフィンガープリント

各ルートCA証明書には一意のフィンガープリントがあります。これは証明書の内容のハッシュ値です。ルートCA証明書のフィンガープリントを使用して、ルートCAの有効性を認証できます。

証明書失効リスト

証明書失効リスト(CRL)は、失効した証明書のシリアル番号のリストです。CRLは作成され、最初に証明書を発行したCAによって署名されます。

CAは定期的にCRLを発行して証明書を失効させます。失効された証明書に関連付けられているエンティティは信頼されません。

次のいずれかの条件が発生した場合、CAは証明書を失効させる必要があります。

- 証明書のサブジェクト名が変更されます。
- 秘密キーが侵害されている。
- サブジェクトとCA間の関連付けが変更されます。たとえば、従業員が組織での雇用を終了する場合などです。

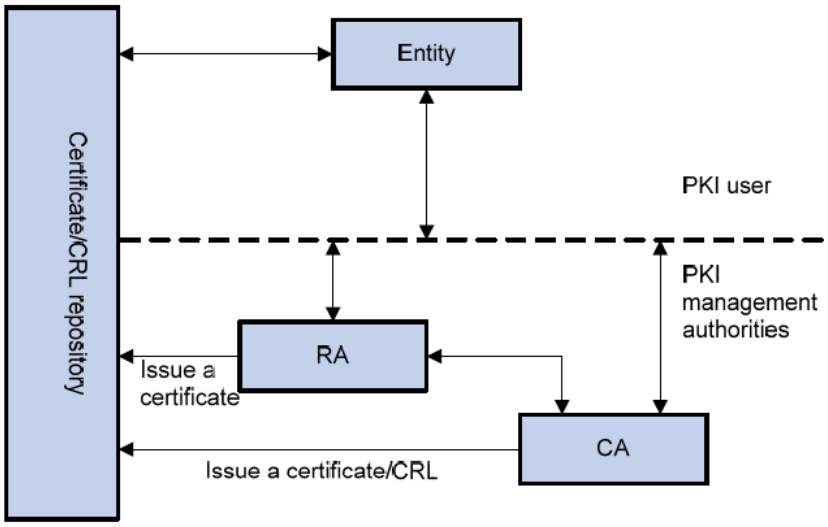
CAポリシー

CAポリシーとは、証明書要求を処理し、証明書を発行および失効させ、CRLを公開するためにCAが従う一連の基準です。通常、CAはCPS(Certification Practice Statement)でポリシーをアドバタイズします。CAポリシーは、電話、ディスク、電子メールなどの帯域外手段を使用して取得できます。CAによって異なるポリシーが使用される場合があるため、証明書要求に対して信頼できるCAを選択する前に、CAポリシーを理解していることを確認してください。

PKIアーキテクチャ

PKIシステムは、図1に示すように、PKIエンティティ、CA、RA、および証明書/CRLリポジトリーから構成されます。

図1 PKIアーキテクチャ



PKIエンティティ

PKIエンティティは、PKI証明書を使用するエンドユーザーです。PKIエンティティには、オペレータ、組織、ルータやスイッチなどのデバイス、またはコンピュータ上で実行されるプロセスなどがあります。PKIエンティティは、SCEPを使用してCAまたはRAと通信します。

CA

認証局(CA)は、証明書を認可および管理します。CAは証明書を発行し、証明書の有効期間を定義し、CRLを発行することによって証明書を取り消します。

RA

登録機関(RA)は、証明書登録要求を処理することによってCAをオフロードします。RAは、証明書要求を受け入れ、ユーザーIDを確認し、CAに証明書の発行を要求するかどうかを決定します。

RAはPKIシステムではオプションです。CAを直接ネットワークアクセスにさらすことにセキュリティ上の懸念がある場合は、いくつかのタスクをRAに委任することをお勧めします。その後、CAは署名証明書とCRLの主要なタスクに集中できます。

証明書/CRLリポジトリー

証明書/CRLリポジトリーは、証明書およびCRLを格納し、これらの証明書およびCRLをPKIエンティティに配布する証明書配布ポイントです。また、クエリー機能も提供します。

PKIリポジトリーは、LDAPまたはHTTPプロトコルを使用するディレクトリサーバーにすることができます。LDAPは一般的に使用されます。

デジタル証明書の取得、使用、および保守

次のワークフローでは、デジタル証明書の取得、使用および保守について説明します。この例では、RAを持つCAを使用して、証明書登録要求を処理します。

1. PKIエンティティは、非対称キーペアを生成し、証明書要求をRAに送信します。証明書要求には、公開鍵とそのID情報が含まれます。
2. RAはエンティティのアイデンティティを検証し、アイデンティティ情報と公開鍵を含むデジタル署名をCAに送信します。
3. CAはデジタル署名を確認し、要求を承認して、証明書を発行します。
4. CAから証明書を受信した後、RAは証明書を証明書リポジトリーに送信し、証明書が発行されたことをPKIエンティティに通知します。
5. PKIエンティティは、証明書リポジトリーから証明書を取得します。
6. 通信のためのセキュアな接続を確立するために、2つのPKIエンティティはローカル証明書を交換して相互に認証します。接続は、両方のエンティティがピアの証明書が有効であることを確認した場合にのみ確立できます。
7. 次のいずれかの状況が発生した場合は、PKIエンティティのローカル証明書を削除して、新しい証明書を要求できます。
   - ローカル証明書が期限切れになります。
   - 証明書の秘密キーが侵害されています。

PKIアプリケーション

PKI技術は、オンライントランザクションのセキュリティ要件を満たすことができます。インフラストラクチャとして、PKIには幅広いアプリケーションがあります。H3CのPKIシステムは、IPSecおよびSSLの証明書管理を提供できます。

次に、いくつかの適用例を示します。

VPN

VPNは、公衆通信インフラストラクチャ上に構築されたプライベートデータ通信ネットワークです。VPNでは、PKIベースの暗号化およびデジタル署名テクノロジーとともにネットワーク層セキュリティプロトコル(IPSecなど)を使用して、機密性を確保できます。

セキュア電子メール

PKIは、機密性、整合性、認証、および否認防止に関する電子メール要件に対応できます。一般的なセキュア電子メールプロトコルは、Secure/Multipurpose Internet Mail Extensions(S/MIME)です。これは、PKIに基づいており、署名付きの暗号化された電子メールの転送を可能にします。

Webセキュリティ

PKIは、SSLハンドシェイクフェーズで使用して、デジタル証明書によって通信当事者のIDを検証できます。

PKIタスクの概要

PKIを設定するには、次の作業を実行します。

1. PKIエンティティの設定
2. PKIドメインの設定
3. (任意)証明書およびCRLの格納パスの指定
4. 証明書の要求

   次のいずれかのタスクを選択します。

   - 自動オンライン証明書要求モードを有効にする
   - オンライン証明書要求を手動で送信する
   - オフラインモードで証明書要求を手動で送信する
5. (任意)証明書要求の中断
6. (オプション)証明書の取得

   PKIドメインに関連するCA証明書、ローカル証明書、およびピア証明書をCAから取得し、ローカルに保存して、検索効率を高めることができます。
7. (任意)PKI証明書の確認
8. (オプション)証明書のエクスポート
9. (任意)証明書の削除
10. (任意)証明書ベースのアクセスコントロールポリシーの設定

    証明書ベースのアクセス制御ポリシーを使用すると、認証されたクライアントの証明書の属性に基づいて、デバイス(たとえば、HTTPSサーバー)へのアクセスを認可できます。

PKIエンティティの設定

PKIエンティティについて

証明書申請者は、エンティティを使用して、そのID情報をCAに提供する。有効なPKIエンティティは、次のIDカテゴリの1つ以上を含まなければならない。

- エンティティの識別名(DN)。これには、共通名、国コード、地域、組織、組織の単位および状態が含まれます。エンティティのDNを構成する場合は、共通名が必要です。
- エンティティのFQDN。
- エンティティのIPアドレス。

PKIエンティティ設定に関する制約事項およびガイドライン

PKIエンティティを設定する場合は、次の制約事項およびガイドラインに従ってください。

- アイデンティティカテゴリが必須かオプションかは、CAポリシーによって異なります。CAポリシーに従ってエンティティ設定を構成します。たとえば、CAポリシーでエンティティDNが必要であるが、IPアドレスのみを構成する場合、CAはエンティティからの証明書リクエストを拒否します。
- Windows2000CAサーバー上のSCEPアドオンには、証明書リクエストのデータ長に関する制限があります。PKIエンティティからのリクエストがデータ長の制限を超えた場合、CAサーバーは証明書リクエストに応答しません。この場合、アウトオブバンド手段を使用してリクエストを送信できます。RSAサーバーやOpenCAサーバーなどの他のタイプのCAサーバーには、このような制限はありません。

PKIエンティティタスクの概要

PKIエンティティを設定するには、次の作業を実行します。

1. PKIエンティティのDNの設定
2. PKIエンティティのFQDNの設定
3. PKIエンティティのIPアドレスの設定

PKIエンティティのDNの設定

PKIエンティティのDNを設定するための制約事項およびガイドライン

PKIエンティティのDN文字列構築するために使用される個々のアトリビュートを設定することも、subject-dnコマンドを使用して完全なDNストリングを直接設定することもできます。

subject-dnコマンドが設定されている場合、common-name、country、locality、organization、organization-unit、およびstateコマンドを使用して設定された個々のDNアトリビュートは有効になりません。

個々のDN属性の構成

1. システムビューに入ります。
   system-view
2. PKIエンティティを作成し、そのビューに入ります。
   pki entity entity-name
3. エンティティの共通名を設定します。
   common-name common-name-sting
   デフォルトでは、共通名は設定されません。
4. エンティティの国コードを設定します。
   country country-code-string
   デフォルトでは、国コードは設定されません。
5. エンティティの局所性を設定します。
   locality locality-name
   デフォルトでは、局所性は設定されません。
6. エンティティの組織を設定します。
   organization org-name
   デフォルトでは、組織は設定されていません。
7. 組織のエンティティの単位を設定します。
   organization-unit org-unit-name
   デフォルトでは、単位は設定されません。
8. エンティティが存在する状態を設定します。
   state state-name
   デフォルトでは、状態は設定されていません。

完全なDNストリングの構成

1. システムビューに入ります。
   system-view
1. PKIエンティティを作成し、そのビューに入ります。
   pki entity entity-name
2. 完全なサブジェクトDN文字列を設定します。
   subject-dn dn-string
   デフォルトでは、PKIエンティティDNは設定されていません。

PKIエンティティのFQDNの設定

1. システムビューに入ります。
   system-view
2. PKIエンティティを作成し、そのビューに入ります。
   pki entity entity-name
3. PKIエンティティのFQDNを設定します。
   fqdn fqdn-name-string
   デフォルトでは、FQDNは設定されていません。

PKIエンティティのIPアドレスの設定

1. システムビューに入ります。
   system-view
2. PKIエンティティを作成し、そのビューに入ります。
   pki entity entity-name
3. PKIエンティティのIPアドレスを設定します。
   ip { ip-address | interface interface-type interface-number }
   デフォルトでは、IPアドレスは設定されていません。

PKIドメインの設定

PKIドメインについて

PKIドメインには、PKIエンティティの登録情報が含まれます。これはローカルに重要であり、IKEやSSLなどの他のアプリケーションでの使用のみを目的としています。

PKIドメインタスクの概要

PKIドメインを設定するには、次の作業を実行します。

1. PKIドメインの作成
2. トラステッドCAの指定
3. PKIエンティティ名の指定
4. 証明書要求受信権限の指定
5. 証明書要求のURLの指定
6. (任意)SCEPポーリング間隔および最大ポーリング試行回数の設定
7. LDAPサーバーの指定

このタスクは、次のいずれかの条件が満たされる場合に必要です。

- デバイスは、LDAPプロトコルを使用してCAから証明書を取得する必要があります。
- LDAPサーバーのホスト名を含まないLDAP URLは、CRLリポジトリーURLとして指定されます。

8. ルートCA証明書検証のためのフィンガープリントの指定

   この手順は、自動証明書要求モードがPKIドメインで設定されている場合に必要です。

   手動証明書要求モードが設定されている場合は、この手順を省略して、ルートCA証明書の検証中に表示されるフィンガープリントを手動で検証できます。
9. 証明書要求のキーペアの指定
10. (任意)証明書の使用目的の指定
11. (任意)PKIプロトコルパケットの送信元IPアドレスの指定
12. (任意)PKCS#7形式の証明書ファイルの暗号化アルゴリズムの指定

PKIドメインの作成

1. システムビューに入ります。

   system-view
2. PKIドメインを作成し、そのビューに入ります。

   pki domain domain-name

トラステッドCAの指定

このタスクについて

PKIドメインには、ローカル証明書を要求する前にCA証明書が必要です。CA証明書を取得するには、信頼できるCA名を指定する必要があります。信頼できるCA名は、CAサーバーに複数のCAが存在する場合に使用されるCAを一意に識別します。

手順

1. システムビューに入ります。

   system-view
2. PKIドメインビューに入ります。

   pki domain domain-name
3. 信頼できるCAの名前を指定します。

   ca identifier name

   デフォルトでは、信頼できるCA名は指定されていません。

PKIエンティティ名の指定

1. システムビューに入ります。

   system-view
2. PKIドメインビューに入ります。

   pki domain domain-name
3. PKIエンティティ名を指定します。

   certificate request entity entity-name

   デフォルトでは、PKIエンティティ名は指定されません。

証明書要求受信権限の指定

1. システムビューに入ります。
   system-view
2. PKIドメインビューに入ります。
   pki domain domain-name
3. 証明書要求の受信機関を指定します。
   certificate request from { ca | ra }
   デフォルトでは、証明書要求の受信機関は指定されていません。

証明書要求のURLの指定

1. システムビューに入ります。
   system-view
2. PKIドメインビューに入ります。
   pki domain domain-name
3. デバイスが証明書要求を送信する証明書要求受信機関のURLを指定します。
   certificate request url url-string
   デフォルトでは、証明書要求のURLは指定されません。

SCEPポーリング間隔および最大ポーリング試行回数の設定

1. システムビューに入ります。
   system-view
2. PKIドメインビューに入ります。
   pki domain domain-name
3. SCEPポーリング間隔およびポーリング試行の最大回数を設定します。
   certificate request polling { count count | interval interval }
   デフォルトでは、デバイスは20分ごとにCAサーバーに対して証明書リクエストのステータスをポーリングします。ポーリングの最大試行回数は50回です。

LDAPサーバーの指定

1. システムビューに入ります。
   system-view
2. PKIドメインビューに入ります。
   pki domain domain-name
3. LDAPサーバーを指定します。
   ldap-server host hostname [port port-number]
   デフォルトでは、LDAPサーバーは指定されていません。

ルートCA証明書検証のためのフィンガープリントの指定

1. システムビューに入ります。
   system-view
2. PKIドメインビューを入力します。
   pki domain domain-name
3. ルートCA証明書を確認するためのフィンガープリントを設定します。
   root-certificate fingerprint { md5 | sha1 } string
   デフォルトでは、フィンガープリントは設定されていません。

証明書要求のキーペアの指定

このタスクについて

PKIドメイン内の証明書要求には、次のいずれかのタイプのキーペアを指定できます。

- DSA。
- ECDSA。
- RSA。

鍵ペアの秘密鍵は秘密に保たれます。鍵ペアの公開鍵は、証明書要求内の他の情報とともにCAに送信されます。CAは要求データに署名して証明書を発行します。

DSA、ECDSA、およびRSAキーペアの詳細については、「公開鍵の管理」を参照してください。

制限事項およびガイドライン

存在しないキーペアを証明書要求に指定できます。PKIエンティティは、証明書要求を送信する前に、キーペアを自動的に作成します。

手順

1. システムビューに入ります。
   system-view
2. PKIドメインビューを入力します。
   pki domain domain-name
3. 証明書要求のキーペアを指定します。
   - RSAキーペアを指定します。
     public-key rsa { { encryption name encryption-key-name [length key-length] | signature name signature-key-name [length key-length] } * | general name key-name [length key-length] }
   - ECDSAキーペアを指定します。
     public-key ecdsa name key-name [secp192r1 | secp256r1 | secp384r1 | secp521r1]
   - DSA鍵ペアを指定します。
     public-key dsa name key-name [length key-length]

   デフォルトでは、キーペアは指定されていません。

証明書の意図された目的の指定

このタスクについて

発行された証明書には、証明書の使用を特定の目的に制限する拡張が含まれています。CAに送信される証明書要求に含まれる証明書の目的を指定できます。ただし、発行された証明書に含まれる実際の拡張はCAポリシーに依存し、PKIドメインで指定されている拡張とは異なる場合があります。アプリケーション(IKEやSSLなど)が認証中に証明書を使用するかどうかは、アプリケーションのポリシーによって決まります。

サポートされる証明書拡張には、次のものがあります

- ike: この拡張を伝送する証明書は、IKEピアによって使用できます。
- ssl-client: この拡張機能を持つ証明書は、SSLクライアントで使用できます。
- ssl-server: この拡張機能を持つ証明書は、SSLサーバーで使用できます。

手順

1. システムビューに入ります。

   system-view
2. PKIドメインビューに入ります。

   pki domain domain-name
3. 証明書の使用目的を指定します。

   usage { ike | ssl-client | ssl-server } *

   デフォルトでは、証明書は、IKE、SSLクライアント、およびSSLサーバーなど、サポートされるすべてのアプリケーションで使用できます。

PKIプロトコルパケットの送信元IPアドレスの指定

このタスクについて

このタスクは、CAポリシーでCAサーバーが特定のIPアドレスまたはサブネットからの証明書要求を受け入れる必要がある場合に必要です。

手順

1. システムビューに入ります。

   system-view
2. PKIドメインビューに入ります。

   pki domain domain-name
3. PKIプロトコルパケットの送信元IPアドレスを指定します。

   IPv4:

   source ip { ip-address | interface interface-type interface-number }

   IPv6:

   source ipv6 { ipv6-address | interface interface-type interface-number }

   デフォルトでは、PKIプロトコルパケットの送信元IPアドレスは、発信インターフェイスのIPアドレスです。

PKCS#7形式の証明書ファイルの暗号化アルゴリズムの

指定

このタスクについて

オンライン証明書リクエスト中、デバイスは指定された暗号化アルゴリズムを使用して、PKCS#7形式で証明書署名リクエストを暗号化してから、リクエストをCAに送信します。CAによって発行された証明書を取得した後、デバイスは暗号化アルゴリズムを使用してPKCS#7形式で証明書ファイルを復号化します。指定された暗号化アルゴリズムがCAサーバーでサポートされていることを確認してください。

手順

1. システムビューに入ります。

   system-view
2. PKIドメインビューに入ります。

   pki domain domain-name
3. PKCS#7形式の証明書ファイルの暗号化アルゴリズムを指定します。

   pkcs7-encryption-algorithm { 3des-cbc | aes-cbc-128 | des-cbc }

   デフォルトでは、DES-CBC暗号化アルゴリズムが使用されます。

証明書およびCRLの格納パスの指定

このタスクについて

デバイスには、証明書およびCRLのデフォルトの格納パスがあります。格納パスを変更して、証明書およびCRLに異なるパスを指定できます。

証明書またはCRLの格納パスを変更した後、元のパスの証明書ファイルとCRLファイルは新しいパスに移動されます。証明書ファイルにはファイル拡張子.cerまたは.p12が使用され、CRLファイルにはファイル拡張子.crlが使用されます。

制限事項およびガイドライン

ストレージパスを変更した場合は、デバイスを再起動またはシャットダウンする前に設定を保存して、証明書またはCRLが失われないようにしてください。

手順

1. システムビューに入ります。

   system-view
2. 証明書およびCRLの保存パスを指定します。

   pki storage { certificates | crls } dir-path

   デフォルトでは、デバイスは証明書とCRLをデバイスのストレージメディア上のPKIディレクトリに格納します。

証明書の要求

証明書要求の構成について

証明書を要求するには、PKIエンティティは、その識別情報と公開鍵をCAに提供する必要があります。

証明書要求は、オフラインモードまたはオンラインモードでCAに送信できます。

- **オフラインモード**: 証明書要求は、次のようなアウトオブバンド方式を使用して送信されます。電話、ディスク、または電子メール。
- **オンラインモード** :証明書要求は、Simple Certificate Enrollment Protocol(SCEP)を使用して、自動または手動でCAに送信できます。

証明書要求の設定に関する制約事項およびガイドライン

PKIドメイン内のローカル証明書を要求する場合は、次の制約事項およびガイドラインに従ってください。

- 既存のローカル証明書が無効にならないようにするには、次のタスクを実行しないでください。
  - 証明書に含まれているキーペアと同じ名前のキーペアを作成します。キーペアを作成するには、public-key local createコマンドを使用します。
  - 証明書に含まれているキーペアを破棄します。
    キーペアを破棄するには、public-key local destroyコマンドを使用します。

  public-key local createコマンド公開鍵ローカル破棄コマンドの詳細については、「セキュリティコマンドリファレンス」の「公開鍵管理コマンド」を参照してください。
- すでにローカル証明書を持つPKIドメイン内の新しい証明書を手動で要求するには、次の手順を実行します。
  a. 既存のローカル証明書を削除するには、**pki delete-certificate**コマンドを使用します。
  b. 新しいキーペアを生成するには、**public-key local create**コマンドを使用します。
  c. 証明書要求を手動で送信します。
- PKIドメインは、1種類の暗号アルゴリズム(DSA、ECDSA、またはRSA)のみを使用するローカル証明書を持つことができます。DSAまたはECDSAが使用される場合、PKIドメインは1つのローカル証明書のみを持つことができます。RSAが使用される場合、PKIドメインは1つのローカル証明書を署名用に、1つのローカル証明書を暗号化用に持つことができます。

証明書要求設定の前提条件

デバイスがCAサーバーと時間同期していることを確認します。デバイスがCAサーバーと時間同期していない場合、証明書が有効期間外であるとみなされるため、証明書要求が失敗する場合があります。システム時間の構成の詳細は、『システム管理コンフィギュレーションガイド』の「システム管理」を参照してください。

自動オンライン証明書要求モードを有効にする

このタスクについて

自動要求モードでは、ローカル証明書を持たないPKIエンティティは、アプリケーションがPKIエンティティと連携するときに、自動的に証明書要求をCAに送信します。たとえば、IKEネゴシエーションでID認証にデジタル署名が使用されているが、使用可能なローカル証明書がない場合、エンティティは自動的に証明書要求を送信します。CAから証明書を取得した後、証明書をローカルに保存します。

CA証明書は、ローカル証明書を要求する前に存在する必要があります。PKIドメインにCA証明書が存在しない場合、PKIエンティティは、証明書要求を送信する前に自動的にCA証明書を取得します。

証明書の自動更新を使用すると、古い証明書の有効期限が切れる前の指定した日数に新しい証明書を自動的に要求できます。古い証明書は、新しい証明書を受信するとすぐに置き換えられます。

制限事項およびガイドライン

自動要求モードでは、現在の証明書が期限切れになるか期限切れになった場合、デバイスは新しい証明

書を自動的に要求しません。これによりサービスが中断される可能性があります。

証明書の期限切れによるサービスの中断を回避するには、**renew-before-expire days**オプションを指定して、自動証明書要求モードで証明書の自動更新をイネーブルにします。

一部のCAでは、証明書の自動更新を機能させるために、新しいPKIエンティティの共通名が必要です。**automatic-append common-name**キーワードを使用すると、証明書の自動更新が成功します。

手順

1. システムビューに入ります。

   system-view
2. PKIドメインビューに入ります。

   pki domain domain-name
3. 自動オンライン証明書要求モードをイネーブルにします。

   certificate request mode auto [password { cipher | simple } string | renew-before-expire days [reuse-public-key] [automatic-append common-name]] *

   デフォルトでは、手動要求モードが適用されます。

   CAポリシーで証明書失効用のパスワードが必要な場合は、このコマンドでパスワードを指定します。

オンライン証明書要求を手動で送信する

このタスクについて

手動要求モードでは、pki request-certificate domainコマンドを実行して、PKIドメイン内のローカル証明書を要求する必要があります。証明書は、CAから取得された後、ドメインに保存されます。

手順

1. システムビューに入ります。

   system-view
2. PKIドメインビューを入力します。

   pki domain domain-name
3. 証明書要求モードを手動に設定します。証明書要求モード手動デフォルトでは、手動要求モードが適用されます。
4. システムビューに戻ります。

   quit
5. CA証明書を取得します。

   「証明書の取得」を参照してください。

   このステップは、PKIドメインがCA証明書を持っていない場合に必要です。CA証明書は、取得されたローカル証明書の信頼性と妥当性を検証するために使用されます。
6. SCEP証明書要求を手動で送信します。

   pki request-certificate domain domain-name [password password]

   このコマンドはコンフィギュレーションファイルには保存されません。

   CAポリシーで証明書失効用のパスワードが必要な場合は、このコマンドでパスワードを指定します。

オフラインモードで証明書要求を手動で送信する

このタスクについて

CAがSCEPをサポートしていない場合、またはデバイスとCA間のネットワーク接続が不可能な場合は、この方法を使用します。

手順

1. システムビューに入ります。

   system-view
2. PKIドメインビューに入ります。

   pki domain domain-name
3. 証明書要求モードを手動に設定します。証明書要求モード手動デフォルトでは、手動要求モードが適用されます。
4. システムビューに戻ります。

   quit
5. CA証明書を取得します。

   「証明書の取得」を参照してください。

   このステップは、PKIドメインがCA証明書を持っていない場合に必要です。CA証明書は、取得されたローカル証明書の信頼性と妥当性を検証するために使用されます。
6. 証明書要求を端末上でPKCS10形式で印刷するか、証明書要求をPKCS10ファイルに保存します。

   pki request-certificate domain domain-name pkcs10 [filename *filename*]

   このコマンドはコンフィギュレーションファイルには保存されません。
7. アウトオブバンド方式を使用して、証明書要求情報をCAに転送します。
8. アウトオブバンド方式を使用して、発行されたローカル証明書をCAからローカルデバイスに転送します。
9. ローカル証明書をPKIドメインにインポートします。

   pki import domain domain-name { der local filename *filename* **| p12 local filename** *filename* **| pem local } [filename** *filename* **] }**

証明書要求の中断

このタスクについて

CAが証明書を発行する前に、証明書要求を中断して、共通名、国コード、FQDNなどのパラメータを変更できます。証明書要求のステータスを表示するには、**display pki certificate request-status**コマンドを使用できます。

または、PKIドメインを削除して、関連する証明書要求を中断することもできます。

手順

1. システムビューに入ります。

   system-view
2. 証明書要求を中止します。

   pki abort-certificate-request domain domain-name

このコマンドはコンフィギュレーションファイルには保存されません。

証明書の取得

このタスクについて

PKIドメインに関連するCA証明書、ローカル証明書、およびピア証明書をCAから取得し、ローカルに保存して、検索効率を向上させることができます。そのためには、オフラインモードまたはオンライモードのいずれかを使用します。

- オフラインモードでは、FTP、ディスクまたは電子メールなどのアウトオブバンド手段で証明書を取得し、ローカルにインポートします。このモードは、CRLリポジトリーが指定されていない場合、CAサーバーがSCEPをサポートしていない場合、またはCAサーバーが証明書のキーペアを生成する場合に使用します。
- オンラインモードでは、SCEPを介してCA証明書を取得し、LDAPを介してローカル証明書またはピア証明書を取得できます。

制限事項およびガイドライン

CAから証明書を取得する場合は、次の制限事項およびガイドラインに従ってください。

- CA証明書がすでにローカルに存在する場合、オンラインモードで再度取得することはできません。新しいCA証明書を取得する場合は、まず、pki delete-certificateコマンドを使用して既存のCA証明書およびローカル証明書を削除します。
- ローカルまたはピア証明書がすでに存在する場合は、新しいローカルまたはピア証明書を取得して既存の証明書を上書きできます。RSAが使用されている場合、PKIドメインは2つのローカル証明書(1つは署名用、もう1つは暗号化用)を持つことができます。
- CRLチェックが有効になっている場合、証明書を取得するとCRLチェックがトリガーされます。取得する証明書が取り消されている場合、その証明書は取得できません。
- デバイスは、証明書の有効期間をローカルシステム時間と比較して、証明書が有効かどうかを判別します。デバイスのシステム時間がCAサーバーと同期していることを確認してください。

前提条件

- オンラインモードでローカルまたはピア証明書を取得する前に、LDAPサーバーがPKIドメイン内で正しく設定されていることを確認してください。
- オフラインモードで証明書をインポートする前に、次のタスクを完了してください。
  - FTPまたはTFTPを使用して、証明書ファイルをデバイスのストレージメディアにアップロードします。

    FTPまたはTFTPが使用できない場合は、証明書の内容を表示し、デバイス上のファイルにコピーします。インポートできるのはPEM形式の証明書だけであるため、証明書がPEM形式であることを確認してください。
  - ローカル証明書またはピア証明書をインポートする前に、証明書に署名するCA証明書チェーンを入手します。

    この手順が必要になるのは、CA証明書チェーンがPKIドメインで使用できず、インポートされる証明書にも含まれていない場合だけです。
  - 暗号化されたキーペアを含むローカル証明書をインポートする前に、CA管理者に問い合せて、証明書のインポートに必要なパスワードを取得してください。

手順

1. システムビューに入ります。

   system-view
2. 証明書を取得します。

- オフラインモードで証明書をインポートします。

  pki import domain domain-name { der { ca | local | peer } filename *filename* **| p12 local filename** *filename* **| pem { ca | local | peer } [filename** *filename* **] }**
- オンラインモードで証明書を取得します。

  pki retrieve-certificate domain domain-name { ca | local | peer *entity-name* ***}***

  このコマンドはコンフィギュレーションファイルには保存されません。

PKI証明書の確認

証明の検証について

証明書は、アプリケーションによって要求、取得、または使用されるときに自動的に検証されます。証明書の有効期限が切れた場合、証明書が信頼できるCAによって発行されていない場合、または証明書が取り消された場合、その証明書は使用できません。証明書を手動で検証することもできます。

PKIドメインでCRLチェックを有効または無効にできます。CRLチェックでは、証明書がCRLに含まれているかどうかがチェックされます。含まれている場合、証明書は失効しており、ホームエンティティは信頼されていません。

CRLチェックを使用するには、CRLリポジトリーからCRLを取得する必要があります。デバイスは、次の順序でCRLリポジトリーを選択します。

1. crl urlコマンドを使用してPKIドメインで指定されたCRLリポジトリー。
2. 検証中の証明書内のCRLリポジトリー。
3. 検証される証明書がCA証明書である場合には、CA証明書中のCRLリポジトリーまたは上位レベルCA証明書中のCRLリポジトリー

選択プロセスの後にCRLリポジトリーが見つからない場合、デバイスはSCEPを介してCRLを取得します。このシナリオでは、CA証明書およびローカル証明書が取得されている必要があります。

証明書は、次の状況でCRLチェックに失敗します。

- 証明書のCRLチェック中にCRLを取得できません。
- CRLチェックは、証明書が取り消されたことを確認します。

証明書検証に関する制約事項およびガイドライン

PKIドメインのCA証明書を検証する場合、システムはCA証明書チェーン内のすべての証明書を検証する必要があります。証明書の検証プロセスが正常に行われるようにするには、証明書チェーン内のCA証明書が属するすべてのPKIドメインがデバイスに必要です。

システムは、CA証明書チェーン内のCA証明書を次のように検証します。

1. 最下位レベルの証明書の親証明書を識別します。

   各CA証明書には、証明書を発行した親CAを識別する発行者フィールドが含まれます。
2. 親証明書が属するPKIドメインを特定します。
3. PKIドメインでCRLチェックを実行して、親証明書が失効しているかどうかをチェックします。失効している場合、その証明書は使用できません。

   PKIドメインでCRLチェックがディセーブルになっている場合、この手順は実行されません。
4. ルートCA証明書に到達するまで、CA証明書チェーン内の上位レベル証明書について前述のステップを繰り返します。

5. 証明書チェーン内の各CA証明書が、ルートCAから始まる名前付きの親CAによって発行されていることを確認します。

CRL検査による証明書の検証

制限事項およびガイドライン

revocation-check method noneコマンドがPKIドメインで設定されている場合、CRLチェックは有効になりません。

手順

1. システムビューに入ります。
   system-view
2. PKIドメインビューに入ります。
   pki domain domain-name
3. (任意)CRLリポジトリーのURLを指定します。
   crl url url-string
   デフォルトでは、CRLリポジトリーのURLは指定されません。
4. CRLチェックをイネーブルにします。
   crl check enable
   デフォルトでは、CRLチェックはイネーブルです。
5. システムビューに戻ります。
   quit
6. CA証明書を取得します。
   「証明書の取得」を参照してください。
   PKIドメイン内の証明書を確認するには、PKIドメインにCA証明書が必要です。
7. (任意)CRLを取得し、ローカルに保存します。
   pki retrieve-crl domain domain-name
   非ルートCA証明書およびローカル証明書を検証するために、PKIドメインにCRLがない場合、デバイスは自動的にCRLを取得します。
   新しく取得されたCRLは、古いCRLがある場合には、それを上書きする。
   取得されたCRLは、PKIドメインに格納されたCA証明書チェーン内のCAによって発行される。
8. 証明書の有効性を手動で確認します。
   pki validate-certificate domain domain-name { ca | local }

CRLチェックを行わない証明書の検証

1. システムビューに入ります。
   system-view
2. PKIドメインビューに入ります。
   pki domain domain-name
3. CRLチェックを無効にします。
   undo crl check enable

デフォルトでは、CRLチェックはイネーブルです。

4. システムビューに戻ります。

   quit

5. PKIドメインのCA証明書を取得します。

   「証明書の取得」を参照してください。

   PKIドメイン内の証明書を確認するには、PKIドメインにCA証明書が必要です。

6. 証明書の有効性を手動で確認します。

   pki validate-certificate domain domain-name { ca | local }

   このコマンドはコンフィギュレーションファイルには保存されません。

証明書のエクスポート

このタスクについて

PKIドメイン内のCA証明書およびローカル証明書を証明書ファイルにエクスポートできます。エクスポートされた証明書ファイルは、デバイスまたは他のPKIアプリケーションにインポートして戻すことができます。

制限事項およびガイドライン

すべての証明書をPKCS12形式でエクスポートするには、PKIドメインに少なくとも1つのローカル証明書が必要です。PKIドメインにローカル証明書がない場合、PKIドメイン内の証明書はエクスポートできません。

PEM形式で証明書をエクスポートするときにファイル名を指定しない場合、このコマンドは端末上の証明書の内容を表示します。

RSAキーペアを使用してローカル証明書をファイルにエクスポートする場合、証明書ファイル名がコマンドで指定されたファイル名と異なる場合があります。実際の証明書ファイル名は、証明書に含まれるキーペアの目的によって異なります。ファイルの命名規則の詳細は、『Security Command Reference』のpki exportコマンドを参照してください。

手順

1. システムビューに入ります。

   system-view

2. 証明書をエクスポートします。
   - 証明書をDER形式でエクスポートします。

     pki export domain domain-name der { all | ca | local } filename *filename*
   - 証明書をPKCS12形式でエクスポートします。

     pki export domain domain-name p12 { all | local } passphrase p12-key filename *filename*
   - 証明書をPEM形式でエクスポートします。

     pki export domain domain-name pem { { all | local } [{ 3des-cbc | aes-128-cbc | aes-192-cbc | aes-256-cbc | des-cbc } pem-key] | ca } [filename *filename* **]**

証明書を削除する

このタスクについて

次の状況では、PKIドメインから証明書を削除できます。

- 証明書が期限切れになった場合、または期限切れになりそうな場合は、CA証明書、ローカル証明書、またはピア証明書を削除します。
- 証明書の秘密キーが侵害された場合、または既存の証明書を置き換えるために新しいローカル証明書を要求する場合は、ローカル証明書を削除します。

制限事項およびガイドライン

CA証明書を削除すると、ローカル証明書、ピア証明書、およびCRLがドメインから自動的に削除されます。

ローカル証明書を削除して新しい証明書を要求するには、次のタスクを実行します。

1. ローカル証明書を削除します。
2. 既存のローカルキーペアを破棄するには、public-key local destroyコマンドを使用します。
3. 新しいキーペアを生成するには、public-key local createコマンドを使用します。
4. 新しい証明書を要求します。

公開鍵ローカル破棄コマンドおよび公開鍵ローカル作成コマンドの詳細については、「セキュリティコマンドリファレンス」を参照してください。

手順

1. システムビューに入ります。

   system-view
2. 証明書を削除します。

   pki delete-certificate domain domain-name { ca | local | peer [serial *serial-num***]}**

   シリアル番号を指定せずにpeerキーワードを使用した場合、このコマンドはすべてのピア証明書を削除します。

証明書ベースのアクセス制御ポリシーの設定

証明書ベースのアクセス制御ポリシーについて

証明書ベースのアクセス制御ポリシーを使用すると、認証されたクライアントの証明書の属性に基づいて、デバイス(たとえば、HTTPSサーバー)へのアクセスを認可できます。

アクセス制御規則および証明書属性グループ

証明書ベースのアクセスコントロールポリシーは、アクセスコントロールルールのセット(permitまたはdenyステートメント)であり、それぞれが証明書属性グループに関連付けられています。証明書属性グループには複数の属性ルールが含まれており、それぞれが証明書発行者名、サブジェクト名または代替サブジェクト名フィールド内の属性に対する一致基準を定義しています。

証明書照合メカニズム

証明書がアクセス制御ルールに関連付けられた証明書属性グループ内のすべての属性ルールと一致する場合、システムは証明書がアクセス制御ルールと一致すると判断します。このシナリオでは、一致プロセスが停止し、アクセス制御ルールで定義されたアクセス制御アクションが実行されます。

次の条件は、証明書ベースのアクセスコントロールポリシーが証明書の有効性を確認する方法を示しています。

- 証明書がpermitステートメントと一致する場合、証明書は検証に合格します。
- 証明書がdenyステートメントに一致する場合、またはポリシー内のどのステートメントにも一致しない場合、証明書は無効と見なされます。
- ステートメントが存在しない属性グループに関連付けられている場合、または属性グループに属性

ルールがない場合、証明書はステートメントと一致します。

- セキュリティアプリケーション(たとえば、HTTPS)に指定された証明書ベースのアクセスコントロールポリシーが存在しない場合、アプリケーション内のすべての証明書が検証に合格します。

手順

1. システムビューに入ります。

   system-view
2. 証明書属性グループを作成し、そのビューを入力します。

   pki certificate attribute-group group-name
3. 発行者名、サブジェクト名、または代替サブジェクト名のアトリビュート規則を設定します。

   attribute id { alt-subject-name { fqdn | ip } | { issuer-name | subject-name } { dn | fqdn | ip } } { ctn | equ | nctn | nequ} attribute-value

   デフォルトでは、属性ルールは設定されません。
4. システムビューに戻ります。

   quit
5. 証明書ベースのアクセス制御ポリシーを作成し、そのビューに入ります。

   pki certificate access-control-policy policy-name
6. デフォルトでは、証明書ベースのアクセス制御ポリシーは存在しません。
7. 証明書アクセス制御規則を作成します。

   rule [id] { deny | permit } group-name

   デフォルトでは、証明書アクセス制御規則は設定されておらず、すべての証明書が検証に合格できます。

   証明書ベースのアクセス制御ポリシーに対して、複数の証明書アクセス制御規則を作成できます。

PKIの表示コマンドおよびメンテナンスコマンド

任意のビューで表示コマンドを実行します。

タスク	コマンド
証明書ベースのアクセスコントロールポリシー情報を表示します。	**display pki certificate access-control-policy [** *policy-name* **]**
証明書アトリビュートグループ情報を表示します。	**display pki certificate attribute-group [** *group-name* **]**
証明書の内容を表示します。	**display pki certificate domain domain-name { ca | local | peer [serial** *serial-num*] }
証明書の更新ステータスを表示します。	**display pki certificate renew-status [domain** *domain-name* **]**
証明書要求のステータスを表示します。	**display pki certificate request-status [domain** *domain-name* **]**
PKIドメイン内にローカルに格納されたCRLを表示します。	**display pki crl domain domain-name**

PKIの設定例

一般的な制限およびガイドライン

Windowsサーバー、RSA Keon、OpenCAなどのさまざまなソフトウェアアプリケーションを使用して、CAサーバーとして機能させることができます。

WindowsサーバーまたはOpenCAを使用する場合は、Windowsサーバー用のSCEPアドオンをインストールするか、OpenCA用のSCEPをイネーブルにする必要があります。いずれの場合も、PKIドメインを設定するときには、raコマンドからの証明書要求を使用して、証明書要求を受け入れるRAを指定する必要があります。

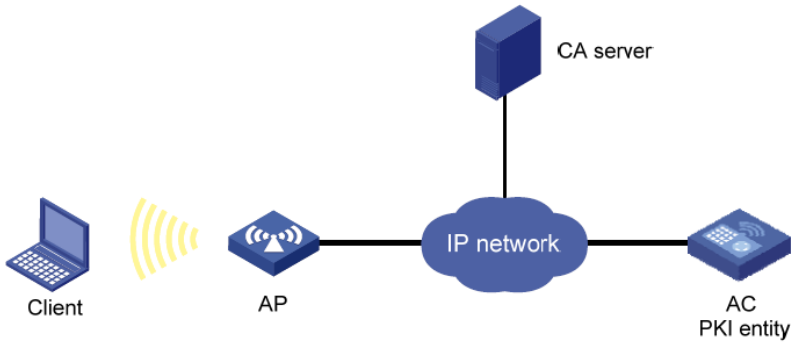
RSA Keonを使用する場合、SCEPアドオンは必要ありません。PKIドメインを設定する場合は、certificate request from caコマンドを使用して、証明書要求を受け入れるCAを指定する必要があります。

例: RSA Keon CAサーバーからの証明書の要求

ネットワーク構成

RSA Keon CAサーバーからローカル証明書を要求するようにPKIエンティティ(AC)を設定します。

図2 ネットワーク図



RSA Keon CAサーバーの設定

1. mycaという名前のCAサーバーを作成します。

   この例では、CAサーバー上で次の基本アトリビュートを設定する必要があります。

   - Nickname: 信頼できるCAの名前。
   - サブジェクトDN: 共通名(CN)、組織単位(OU)、組織(O)、および国(C)を含むCAのDN属性。

   その他の属性にはデフォルト値を使用できます。

2. 拡張属性を構成します。

   CAサーバーの管理ページのJurisdiction Configurationセクションでパラメータを設定します。

   - 正しい拡張プロファイルを選択します。
   - SCEP autovetting機能をイネーブルにして、CAサーバーが手動による介入なしに証明書要求を自動的に承認できるようにします。
   - SCEP autovettingのIPアドレスリストを指定します。

ACの設定

1. デバイスが証明書を正しく要求したり、CRLを取得したりするために、ACのシステム時間をCAサーバーと同期させます(詳細は省略します)。
2. **aaa**という名前のエンティティを作成し、共通名を**AC**に設定します。

   <AC> system-view

   [AC] pki entity aaa

   [AC-pki-entity-aaa] common-name AC

   [AC-pki-entity-aaa] quit
3. PKIドメインを設定します。

   #**torsa**という名前のPKIドメインを作成し、そのビューに入ります。

   [AC] pki domain torsa

   #信頼できるCAの名前を指定します。設定は、CAサーバーに構成されているCA名と同じである必要があります。この例では、**myca**を使用します。

   [AC-pki-domain-torsa] ca identifier myca

   #CAサーバーのURLを構成します。URL形式はhttp://host:port/Issuing管轄区域IDです。ここで、発行管轄区域IDは、CAサーバーで生成される16進数の文字列です。

[AC-pki-domain-torsa] certificate request url
http://1.1.2.22:446/80f6214aa8865301d07929ae481c7ceed99f95bd

ACがcaに証明書を送るように要求するように設定します。

[AC-pki-domain-torsa] certificate request from ca
#PKIエンティティ名をaaaに設定します。
[AC-pki-domain-torsa] certificate request entity aaa
#CRLリポジトリーのURLを指定します。
[AC-pki-domain-torsa] crl url ldap://1.1.2.22:389/CN=myca
#abcという名前の汎用RSAキーペアを1024ビットの長さで設定します。
[AC-pki-domain-torsa] public-key rsa general name abc length 1024
[AC-pki-domain-torsa] quit

4. RSAキーペアを生成します。

[AC] public-key local create rsa name abc
The range of public key modulus is (512 ~ 2048).
If the key modulus is greater than 512,it will take a few minutes. Press
CTRL+C to abort.
Input the modulus length [default = 1024]:
Generating Keys...
.....++++++
.....++++++
Create the key pair successfully.

5. ローカル証明書を要求します。

#CA証明書を取得し、ローカルに保存します。
[AC] pki retrieve-certificate domain torsa ca

The trusted CA's finger print is:
MD5 fingerprint:EDE9 0394 A273 B61A F1B3 0072 A0B1 F9AB
SHA1 fingerprint: 77F9 A077 2FB8 088C 550B A33C 2410 D354 23B2 73A8
Is the finger print correct?(Y/N):y Retrieved
the certificates successfully.
#証明書要求を手動で送信し、証明書失効パスワードを1111に設定します。
RSA Keon CAサーバーを使用する場合は、証明書失効パスワードが必要です。
[AC] pki request-certificate domain torsa password 1111

Start to request certificate...
……
Certificate requested successfully.

設定の確認

#PKIドメインtorsa内のローカル証明書に関する情報を表示します。
[AC]display pki certificate domain torsa local

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

15:79:75:ec:d2:33:af:5e:46:35:83:bc:bd:6e:e3:b8

Signature Algorithm: sha1WithRSAEncryption Issuer:

CN=myca

Validity

Not Before: Jan 6 03:10:58 2013 GMT

Not After : Jan 6 03:10:58 2014 GMT

Subject: CN=AC

Subject Public Key Info:

Public Key Algorithm: rsaEncryption Public-

Key: (1024 bit)

Modulus:

00:ab:45:64:a8:6c:10:70:3b:b9:46:34:8d:eb:1a:
a1:b3:64:b2:37:27:37:9d:15:bd:1a:69:1d:22:0f:
3a:5a:64:0c:8f:93:e5:f0:70:67:dc:cd:c1:6f:7a:
0c:b1:57:48:55:81:35:d7:36:d5:3c:37:1f:ce:16:
7e:f8:18:30:f6:6b:00:d6:50:48:23:5c:8c:05:30:
6f:35:04:37:1a:95:56:96:21:95:85:53:6f:f2:5a:
dc:f8:ec:42:4a:6d:5c:c8:43:08:bb:f1:f7:46:d5:
f1:9c:22:be:f3:1b:37:73:44:f5:2d:2c:5e:8f:40: 3e:36:36:0d:c8:33:90:f3:9b

Exponent: 65537 (0x10001)

X509v3 extensions:

X509v3 CRL Distribution Points:

Full Name:

DirName: CN = myca

Signature Algorithm: sha1WithRSAEncryption b0:9d:d9:ac:a0:9b:83:99:bf:9d:0a:ca:12:99:58:60:d8:aa:
73:54:61:4b:a2:4c:09:bb:9f:f9:70:c7:f8:81:82:f5:6c:af:
25:64:a5:99:d1:f6:ec:4f:22:e8:6a:96:58:6c:c9:47:46:8c:
f1:ba:89:b8:af:fa:63:c6:c9:77:10:45:0d:8f:a6:7f:b9:e8:
25:90:4a:8e:c6:cc:b8:1a:f8:e0:bc:17:e0:6a:11:ae:e7:36:
87:c4:b0:49:83:1c:79:ce:e2:a3:4b:15:40:dd:fe:e0:35:52:
ed:6d:83:31:2c:c2:de:7c:e0:a7:92:61:bc:03:ab:40:bd:69:
1b:f5

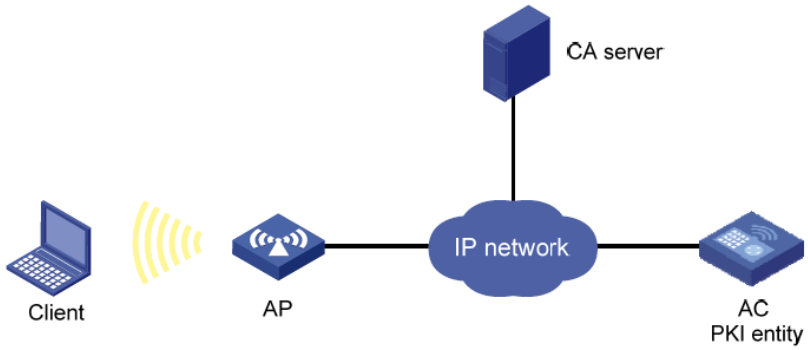
CA証明書に関する詳細情報を表示するには、display pki certificate domainコマンドを使用します。

例: Windows サーバー2003 CAサーバーからの証明書の要求

ネットワーク構成

Windows サーバー2003 CAサーバーからローカル証明書を要求するように、PKIエンティティ(AC)を設定します。

図3 ネットワーク図



Windows Server2003CAサーバーの設定

1. 証明書サービスコンポーネントをインストールします。
   a. Startメニューから**Control Panel > Add or Remove Program**を選択します。
   b. **Add/Remove Windows Components、Certificate Servicesの順に選択します。**
   c. **Next**をクリックして、インストールを開始します。
   d. CA名を設定します。この例では、CA名を**myca**に設定します。
2. SCEPアドオンをインストールします。

   デフォルトでは、Windows Server2003はSCEPをサポートしていません。PKIエンティティを登録してサーバーから証明書を取得するには、サーバーにSCEPアドオンをインストールする必要があります。SCEPアドオンのインストールが完了すると、URLが表示されます。このURLをデバイス上の証明書要求URLとして指定します。
3. 証明書サービスの属性を変更します。
   a. startメニューから **Control Panel > Administrative Tools > Certificate Authority**の順に選択します。

      証明書サービスコンポーネントおよびSCEPアドオンが正常にインストールされている場合は、CAからRAに対して2つの証明書が発行されている必要があります。
   b. ナビゲーションツリーでCAサーバーを右クリックし**Properties>Policy Module**を選択します。
   c. **Properties**をクリックし、**Follow the settings in the certificate template**を選択します(該当する場合)。それ以外の場合は、証明書が自動的に発行されます。
4. インターネットインフォメーションサービスの属性を変更します。
   a. start メニューから**Control Panel > Administrative Tools > Internet Information Services (IIS) Manager**を選択します。
   b. ナビゲーションツリーから**Web Sites**を選択します。
   c. **Default Web Site**を右クリックし、**Properties > Home Directory**を選択します。
   d. **Local path**ボックスに証明書サービスのパスを指定します。
   e. 既存のサービスとの競合を回避するために、デフォルトWebサイトに一意のTCPポート番号を指定します。この例では、ポート8080が使用されています。

ACの設定

1. デバイスが証明書を正しく要求したり、CRLを取得したりするために、ACのシステム時間をCAサーバーと同期させます(詳細は省略します)。
2. **aaa**という名前のエンティティを作成し、共通名を**test**と設定します。

   <AC> system-view

   [AC] pki entity aaa

   [AC-pki-entity-aaa] common-name test

   [AC-pki-entity-aaa] quit
3. Configure a PKI domain:

   #**winserver**という名前のPKIドメインを作成し、そのビューを入力します。

   [AC] pki domain winserver

   #トラステッドCAの名前を**myca**に設定します。

   [AC-pki-domain-winserver] ca identifier myca

   #証明書要求のURLを構成します。URLの形式は **http://host:port/certsrv/mscep/mscep.dll,**です。ここで、host:portはCAサーバーのIPアドレスとポート番号です。

   [AC-pki-domain-winserver] certificate request url http://4.4.4.1:8080/certsrv/mscep/mscep.dll

   #証明書要求を**ra**に送信するようにACを設定します。

   [AC-pki-domain-winserver] certificate request from ra

   #PKIエンティティ名を**aaa**に設定します。

   [AC-pki-domain-winserver] certificate request entity aaa

   #1024ビット長の**abc**という名前の汎用RSAキーペアを設定します。

   [AC-pki-domain-winserver]public-key rsa general name abc length 1024

   [AC-pki-domain-winserver]quit
4. RSAローカルキーペアを生成します。

   [AC] public-key local create rsa name abc

   The range of public key modulus is (512 ~ 2048).

   If the key modulus is greater than 512,it will take a few minutes. Press CTRL+C to abort.

   Input the modulus length [default = 1024]:

   Generating Keys...

   .....++++++

   .....++++++

   Create the key pair successfully.
5. ローカル証明書を要求します。

   #CA証明書を取得し、ローカルに保存します。

   [AC] pki retrieve-certificate domain winserver ca

   The trusted CA's finger print is:

   MD5 fingerprint:766C D2C8 9E46 845B 4DCE 439C 1C1F 83AB

SHA1 fingerprint:97E5 DDED AB39 3141 75FB DB5C E7F8 D7D7 7C9B 97B4

Is the finger print correct?(Y/N):y Retrieved

the certificates successfully. # Submit a

certificate request manually.

[AC] pki request-certificate domain winserver

Start to request certificate ...

…

Certificate requested successfully.

設定の確認

#PKIドメインwinserver内のローカル証明書に関する情報を表示します。

[AC]display pki certificate domain winserver local

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

(Negative)01:03:99:ff:ff:ff:ff:fd:11 Signature

Algorithm: sha1WithRSAEncryption Issuer: CN=sec

Validity

Not Before: Dec 24 07:09:42 2012 GMT

Not After : Dec 24 07:19:42 2013 GMT

Subject: CN=test

Subject Public Key Info:

Public Key Algorithm: rsaEncryption Public-

Key: (2048 bit)

Modulus:

00:c3:b5:23:a0:2d:46:0b:68:2f:71:d2:14:e1:5a:

55:6e:c5:5e:26:86:c1:5a:d6:24:68:02:bf:29:ac:

dc:31:41:3f:5d:5b:36:9e:53:dc:3a:bc:0d:11:fb:

d6:7d:4f:94:3c:c1:90:4a:50:ce:db:54:e0:b3:27:

a9:6a:8e:97:fb:20:c7:44:70:8f:f0:b9:ca:5b:94:

f0:56:a5:2b:87:ac:80:c5:cc:04:07:65:02:39:fc:

db:61:f7:07:c6:65:4c:e4:5c:57:30:35:b4:2e:ed:

9c:ca:0b:c1:5e:8d:2e:91:89:2f:11:e3:1e:12:8a:

f8:dd:f8:a7:2a:94:58:d9:c7:f8:1a:78:bd:f5:42:

51:3b:31:5d:ac:3e:c3:af:fa:33:2c:fc:c2:ed:b9:

ee:60:83:b3:d3:e5:8e:e5:02:cf:b0:c8:f0:3a:a4:

b7:ac:a0:2c:4d:47:5f:39:4b:2c:87:f2:ee:ea:d0:

c3:d0:8e:2c:80:83:6f:39:86:92:98:1f:d2:56:3b:

d7:94:d2:22:f4:df:e3:f8:d1:b8:92:27:9c:50:57:

f3:a1:18:8b:1c:41:ba:db:69:07:52:c1:9a:3d:b1:

2d:78:ab:e3:97:47:e2:70:14:30:88:af:f8:8e:cb:

68:f9:6f:07:6e:34:b6:38:6a:a2:a8:29:47:91:0e:

25:39

Exponent: 65537 (0x10001) X509v3 extensions:

X509v3 Key Usage:

Digital Signature, Non Repudiation, Key Encipherment, Data Encip

X509v3 Subject Key Identifier:
C9:BB:D5:8B:02:1D:20:5B:40:94:15:EC:9C:16:E8:9D:6D:FD:9F:34

X509v3 Authority Key Identifier:
keyid:32:F1:40:BA:9E:F1:09:81:BD:A8:49:66:FF:F8:AB:99:4A:30:21:9

X509v3 CRL Distribution Points:

Full Name: URI:file://\\g07904c\CertEnroll\sec.crl

Authority Information Access:

CA Issuers - URI:http://gc/CertEnroll/gc_sec.crt CA Issuers -
URI:file://\\gc\CertEnroll\gc_sec.crt

1.3.6.1.4.1.311.20.2:

.0.I.P.S.E.C.I.n.t.e.r.m.e.d.i.a.t.e.O.f.f.l.i.n.e Signature Algorithm: sha1WithRSAEncryption

76:f0:6c:2c:4d:bc:22:59:a7:39:88:0b:5c:50:2e:7a:5c:9d:

6c:28:3c:c0:32:07:5a:9c:4c:b6:31:32:62:a9:45:51:d5:f5:

36:8f:47:3d:47:ae:74:6c:54:92:f2:54:9f:1a:80:8a:3f:b2:

14:47:fa:dc:1e:4d:03:d5:d3:f5:9d:ad:9b:8d:03:7f:be:1e:

29:28:87:f7:ad:88:1c:8f:98:41:9a:db:59:ba:0a:eb:33:ec:

cf:aa:9b:fc:0f:69:3a:70:f2:fa:73:ab:c1:3e:4d:12:fb:99:

31:51:ab:c2:84:c0:2f:e5:f6:a7:c3:20:3c:9a:b0:ce:5a:bc:

0f:d9:34:56:bc:1e:6f:ee:11:3f:7c:b2:52:f9:45:77:52:fb:

46:8a:ca:b7:9d:02:0d:4e:c3:19:8f:81:46:4e:03:1f:58:03:

bf:53:c6:c4:85:95:fb:32:70:e6:1b:f3:e4:10:ed:7f:93:27:

90:6b:30:e7:81:36:bb:e2:ec:f2:dd:2b:bb:b9:03:1c:54:0a:

00:3f:14:88:de:b8:92:63:1e:f5:b3:c2:cf:0a:d5:f4:80:47:

6f:fa:7e:2d:e3:a7:38:46:f6:9e:c7:57:9d:7f:82:c7:46:06:

7d:7c:39:c4:94:41:bd:9e:5c:97:86:c8:48:de:35:1e:80:14:

02:09:ad:08

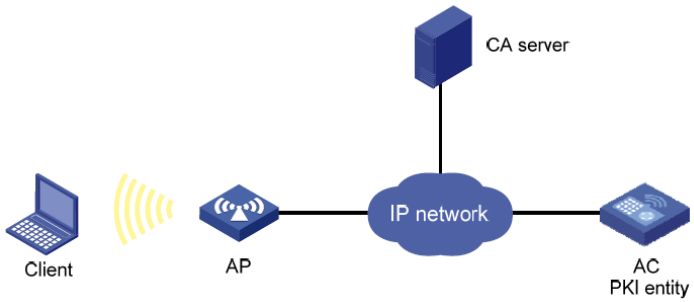
CA証明書に関する詳細情報を表示するには、display pki certificate domainコマンドを使用します。.

例: OpenCAサーバーからの証明書の要求

ネットワーク構成

OpenCA CAサーバーからローカル証明書を要求するようにPKIエンティティ(AC)を設定します。

図4 ネットワーク図



OpenCAサーバーの構成

関連マニュアルの指示に従ってOpenCAサーバーを設定します(詳細は省略します)。

以前のバージョンはSCEPをサポートしていないため、OpenCAサーバーのバージョンがバージョン0.9.2以降であることを確認してください。

ACの設定

1. ACが証明書を正しく要求したりCRLを取得したりするために、ACのシステム時間をCAサーバーと同期させます(詳細は省略します)。
2. **aaa**という名前のPKIエンティティを作成し、そのエンティティの共通名、国コード、組織名、およびOUを設定します。

   <AC> system-view

   [AC] pki entity aaa

   [AC-pki-entity-aaa] common-name rnd

   [AC-pki-entity-aaa] country JP

   [AC-pki-entity-aaa] organization test

   [AC-pki-entity-aaa] organization-unit software

   [AC-pki-entity-aaa] quit
3. PKIドメインを設定します。

   #**openca**という名前のPKIドメインを作成し、そのビューを入力します。

   [AC]pki domain openca

   #トラステッドCAの名前として**myca**を指定します。

   [AC-pki-domain-openca] ca identifier myca

   #証明書要求のURLを設定します。URLの形式はhttp://host/cgi-bin/pki/scepです。ここで、hostはOpenCAサーバーのIPアドレスです。

   [AC-pki-domain-openca] certificate request url http://192.168.222.218/cgi-bin/pki/scep

   #証明書要求をRAに送信するようにACを設定します。

   [AC-pki-domain-openca] certificate request from ra

   #証明書要求にPKIエンティティ**aaa**を指定します。

[AC-pki-domain-openca] certificate request entity aaa

#**abc**という名前の汎用RSAキーペアを1024ビットの長さで設定します。

[AC-pki-domain-openca] public-key rsa general name abc length 1024

[AC-pki-domain-openca] quit

4. RSAキーペアを生成します。

[AC] public-key local create rsa name abc

The range of public key modulus is (512 ~ 2048).

If the key modulus is greater than 512,it will take a few minutes. Press CTRL+C to abort.

Input the modulus length [default = 1024]:

Generating Keys...

.....++++++

.....++++++

Create the key pair successfully.

5. ローカル証明書を要求します。

#CA証明書を取得し、ローカルに保存します。

[AC] pki retrieve-certificate domain openca ca

The trusted CA's finger print is:

MD5 fingerprint:5AA3 DEFD 7B23 2A25 16A3 14F4 C81C C0FA

SHA1 fingerprint:9668 4E63 D742 4B09 90E0 4C78 E213 F15F DC8E 9122

Is the finger print correct?(Y/N): y

Retrieved the certificates successfully.

#証明書要求を手動で送信します。

[AC] pki request-certificate domain openca

Start to request certificate...

…

Certificate requested successfully.

設定の確認

#PKIドメインopenca内のローカル証明書に関する情報を表示します。

[AC] display pki certificate domain openca local

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

21:1d:b8:d2:e4:a9:21:28:e4:de

Signature Algorithm: sha256WithRSAEncryption

Issuer: C=CN, L=shangdi, ST=pukras, O=OpenCA Labs, OU=mysubUnit, CN=sub-ca, DC=pki-subdomain, DC=mydomain-sub, DC=com

Validity

Not Before: Jun 30 09:09:09 2011 GMT

Not After : May 1 09:09:09 2012 GMT

Subject: CN=rnd, O=test, OU=software,

C=CN Subject Public Key Info:

Public Key Algorithm: rsaEncryption

Public-Key: (1024 bit)

Modulus:

00:b8:7a:9a:b8:59:eb:fc:70:3e:bf:19:54:0c:7e:

c3:90:a5:d3:fd:ee:ff:c6:28:c6:32:fb:04:6e:9c:

d6:5a:4f:aa:bb:50:c4:10:5c:eb:97:1d:a7:9e:7d:

53:d5:31:ff:99:ab:b6:41:f7:6d:71:61:58:97:84:

37:98:c7:7c:79:02:ac:a6:85:f3:21:4d:3c:8e:63:

8d:f8:71:7d:28:a1:15:23:99:ed:f9:a1:c3:be:74:

0d:f7:64:cf:0a:dd:39:49:d7:3f:25:35:18:f4:1c:

59:46:2b:ec:0d:21:1d:00:05:8a:bf:ee:ac:61:03:

6c:1f:35:b5:b4:cd:86:9f:45

Exponent: 65537

(0x10001) X509v3 extensions:

X509v3 Basic Constraints:

CA:FALSE

Netscape Cert Type:

SSL Client,

S/MIME

X509v3 Key Usage:

Digital Signature, Non Repudiation, Key Encipherment

X509v3 Extended Key Usage:

TLS Web Client Authentication, E-mail Protection, Microsoft Smartcardlogin

Netscape Comment:

User Certificate of OpenCA Labs

X509v3 Subject Key Identifier:

24:71:C9:B8:AD:E1:FE:54:9A:EA:E9:14:1B:CD:D9:45:F4:B2:7A:1B

X509v3 Authority Key Identifier:

keyid:85:EB:D5:F7:C9:97:2F:4B:7A:6D:DD:1B:4D:DD:00:EE:53:CF:FD:5B

X509v3 Issuer Alternative Name:

DNS:root@docm.com, DNS:, IP Address:192.168.154.145, IP Address:192.168.154.138

Authority Information Access:

CA Issuers - URI:http://192.168.222.218/pki/pub/cacert/cacert.crt OCSP - URI:http://192.168.222.218:2560/

```
                1.3.6.1.5.5.7.48.12 - URI:http://192.168.222.218:830/
            X509v3 CRL Distribution Points:
                Full Name:
                  URI:http://192.168.222.218/pki/pub/crl/cacrl.crl
    Signature Algorithm: sha256WithRSAEncryption
        5c:4c:ba:d0:a1:35:79:e6:e5:98:69:91:f6:66:2a:4f:7f:8b:
        0e:80:de:79:45:b9:d9:12:5e:13:28:17:36:42:d5:ae:fc:4e:
        ba:b9:61:f1:0a:76:42:e7:a6:34:43:3e:2d:02:5e:c7:32:f7:
        6b:64:bb:2d:f5:10:6c:68:4d:e7:69:f7:47:25:f5:dc:97:af:
        ae:33:40:44:f3:ab:e4:5a:a0:06:8f:af:22:a9:05:74:43:b6:
        e4:96:a5:d4:52:32:c2:a8:53:37:58:c7:2f:75:cf:3e:8e:ed:
        46:c9:5a:24:b1:f5:51:1d:0f:5a:07:e6:15:7a:02:31:05:8c:
        03:72:52:7c:ff:28:37:1e:7e:14:97:80:0b:4e:b9:51:2d:50:
        98:f2:e4:5a:60:be:25:06:f6:ea:7c:aa:df:7b:8d:59:79:57:
        8f:d4:3e:4f:51:c1:34:e6:c1:1e:71:b5:0d:85:86:a5:ed:63:
        1e:08:7f:d2:50:ac:a0:a3:9e:88:48:10:0b:4a:7d:ed:c1:03:
        9f:87:97:a3:5e:7d:75:1d:ac:7b:6f:bb:43:4d:12:17:9a:76:
        b0:bf:2f:6a:cc:4b:cd:3d:a1:dd:e0:dc:5a:f3:7c:fb:c3:29:
        b0:12:49:5c:12:4c:51:6e:62:43:8b:73:b9:26:2a:f9:3d:a4:
        81:99:31:89
```

CA証明書に関する詳細情報を表示するには、display pki certificate domainコマンドを使用します。

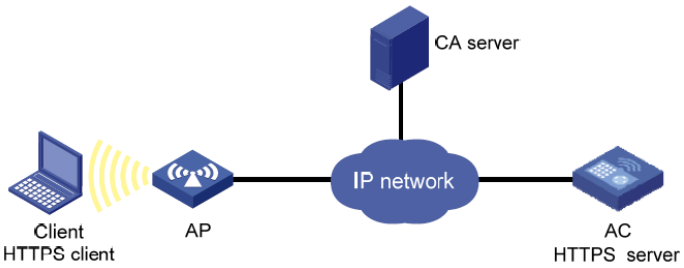
例: 証明書ベースのアクセスコントロールポリシーの設定

ネットワーク構成

図5に示すように、クライアントはHTTPSを介してACにアクセスします。

クライアントを認証し、クライアントの証明書の有効性を確認するために、ACで証明書ベースのアクセス制御ポリシーを設定します。

図5:ネットワーク図



手順

1. SSLで使用されるPKIドメインdomain1を作成します(詳細は省略します)。
2. CAサーバーからACのSSLサーバー証明書を要求します(詳細は省略します)。
3. HTTPSサーバーを設定します。

   #HTTPSサービスのSSLサーバーポリシーを設定します。

   <AC> system-view

   [AC] ssl server-policy abc

   [AC-ssl-server-policy-abc] pki-domain domain1

   [AC-ssl-server-policy-abc] client-verify enable

   [AC-ssl-server-policy-abc] quit

   #SSLサーバーポリシーをHTTPSサービスに適用します。

   [AC] ip https ssl-server-policy abc

   #HTTPSサービスを有効にします。

   [AC] ip https enable
4. 証明書属性グループを構成します。

   #**mygroup1**という名前の証明書属性グループを作成し、2つの属性ルールを追加します。最初のルールは、サブジェクトDNのDNに**abbcc**という文字列が含まれることを定義します。2番目のルールは、証明書発行者のIPアドレスが**10.0.0.1**であることを定義します。

   [AC] pki certificate attribute-group mygroup1

   [AC-pki-cert-attribute-group-mygroup1] attribute 1 subject-name dn ctn aabbcc

   [AC-pki-cert-attribute-group-mygroup1] attribute 2 issuer-name ip equ 10.0.0.1

   [AC-pki-cert-attribute-group-mygroup1] quit

   #**mygroup2**という名前の証明書属性グループを作成し、2つの属性ルールを追加します。最初のルールは、代替サブジェクト名のFQDNに**apple**という文字列が含まれないことを定義します。2番目のルールは、証明書発行者名のDNに**abbcc**という文字列が含まれることを定義します。

   [AC] pki certificate attribute-group mygroup2

[AC-pki-cert-attribute-group-mygroup2] attribute 1 alt-subject-name fqdn nctn apple

[AC-pki-cert-attribute-group-mygroup2] attribute 2 issuer-name dn ctn aabbcc

[AC-pki-cert-attribute-group-mygroup2] quit

5. 証明書ベースのアクセス制御ポリシーを設定します。

#**myacp**という名前の証明書ベースのアクセス制御ポリシーを作成します。

[AC] pki certificate access-control-policy myacp

#証明書属性グループ**mygroup1**の属性規則に一致する証明書を拒否するステートメントを定義します。

[AC-pki-cert-acp-myacp] rule 1 deny mygroup1

#証明書属性グループ**mygroup2**の属性規則に一致する証明書を許可するステートメントを定義します。

[AC-pki-cert-acp-myacp] rule 2 permit mygroup2

[AC-pki-cert-acp-myacp] quit

#証明書ベースのアクセス制御ポリシー**myacp**をHTTPSサービスに適用します。

[AC] ip https certificate access-control-policy myacp

設定の確認

#クライアントで、Webブラウザを使用してHTTPSサーバーにアクセスします。

サーバーは最初に、構成された証明書ベースのアクセス制御ポリシーに従ってクライアントの証明書の妥当性を検証します。クライアントの証明書では、サブジェクトDNは**abbcc**、証明書発行者のIPアドレスは1.1.1.1、代替サブジェクト名のFQDNは**banaba**です。

クライアントの証明書が、証明書ベースのアクセス制御ポリシーの**rule 1**で指定された証明書属性グループ**mygroup1**と一致しません。証明書は**rule 2**との照合を続行します。

クライアントの証明書は、**rule 2**で指定された証明書属性グループ**mygroup2**と一致します。ルール2はpermitステートメントであるため、証明書は検証に合格し、クライアントはHTTPSサーバーにアクセスできます。

例:証明書のインポートとエクスポート

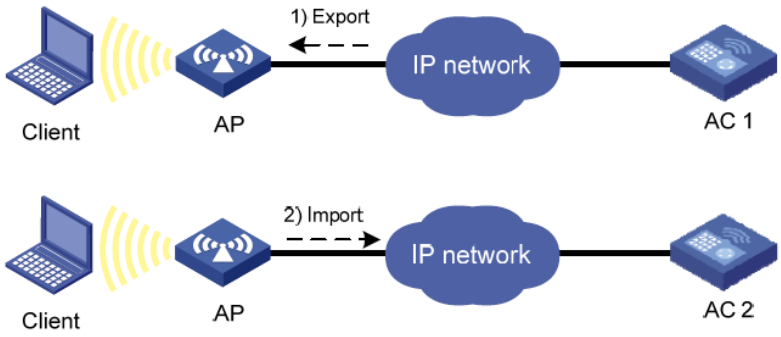
ネットワーク構成

図6に示すように、AC2はネットワーク内のAC1を置き換えます。AC1上のPKIドメイン**exportdomain**には、秘密鍵を含む2つのローカル証明書と1つのCA証明書があります。AC2がAC1を置き換えた後も証明書が有効であることを確認するには、次のようにAC1上の証明書をAC2にコピーします。

1. AC1上のPKIドメイン**exportdomain**の証明書を.pem証明書ファイルにエクスポートします。

   エクスポート中に、3DES_CBCとパスワード11111を使用して、ローカル証明書の秘密キーを暗号化します。
2. 証明書ファイルをFTP経由でAC1からAC2に転送します。
3. 証明書ファイルをAC2上のPKIドメインimportdomainにインポートします。

図6 ネットワーク図



手順

1. 証明書をAC1にエクスポートします。

   #CA証明書を.pemファイルにエクスポートします。

   <AC1> system-view

   [AC1] pki export domain exportdomain pem ca filename pkicachain.pem

   #ローカル証明書を.pemファイルにエクスポートします。ファイル名を**pkilocal.pem**と指定し、3DES_CBCを使用して秘密鍵をパスワード111111で暗号化します。

   [AC1] pki export domain exportdomain pem local 3des-cbc 111111 filename pkilocal.pem

   現在、AC1にはPEM形式の3つの証明書ファイルがあります。

   - **pkicachain.pem**という名前のCA証明書ファイル。
   - **pkilocal.pem-signature**という名前のローカル証明書ファイル。これには、署名用の秘密キーが含まれています。
   - **pkilocal.pem-encryption**という名前のローカル証明書ファイル。暗号化用の秘密キーが含まれます。

   #ローカル証明書ファイル**pkilocal.pem-signature**を表示します。

   [AC1] quit

   <AC1> more pkilocal.pem-signature

   Bag Attributes

   friendlyName:

   localKeyID: 90 C6 DC 1D 20 49 4F 24 70 F5 17 17 20 2B 9E AC 20 F3 99 89

   subject=/C=CN/O=OpenCA Labs/OU=Users/CN=subsign 11

issuer=/C=CN/L=shangdi/ST=pukras/O=OpenCA Labs/OU=docm/CN=subca1

-----BEGIN CERTIFICATE-----

MIIEgjCCA2qgAwIBAgILAJgsebpejZc5UwAwDQYJKoZIhvcNAQELBQAwZjELMAkG

…

-----END CERTIFICATE-----

Bag Attributes

friendlyName

:

localKeyID: 90 C6 DC 1D 20 49 4F 24 70 F5 17 17 20 2B 9E AC 20 F3 99 89

Key Attributes: <No Attributes>

-----BEGIN ENCRYPTED PRIVATE KEY-----

MIICxjBABgkqhkiG9w0BBQ0wMzAbBgkqhkiG9w0BBQwwDgQIZtjSjfslJCoCAggA

…

-----END ENCRYPTED PRIVATE KEY-----

#ローカル証明書ファイルpkilocal.pem暗号化を表示します。

<AC1> more pkilocal.pem-

encryption Bag Attributes

friendlyName:

localKeyID: D5 DF 29 28 C8 B9 D9 49 6C B5 44 4B C2 BC 66 75 FE D6 6C C8

subject=/C=CN/O=OpenCA Labs/OU=Users/CN=subencr 11

issuer=/C=CN/L=shangdi/ST=pukras/O=OpenCA Labs/OU=docm/CN=subca1

-----BEGIN CERTIFICATE-----

MIIEUDCCAzigAwIBAgIKCHxnAVyzWhIPLzANBgkqhkiG9w0BAQsFADBmMQswCQYD

…

-----END CERTIFICATE-----

Bag Attributes

friendlyName

:

localKeyID: D5 DF 29 28 C8 B9 D9 49 6C B5 44 4B C2 BC 66 75 FE D6 6C C8

Key Attributes: <No Attributes>

-----BEGIN ENCRYPTED PRIVATE KEY-----

MIICxjBABgkqhkiG9w0BBQ0wMzAbBgkqhkiG9w0BBQwwDgQI7H0mb4O7/GACAggA

…

-----END ENCRYPTED PRIVATE KEY-----

2. 証明書ファイル**pkicachain.pem**、**pkilocal.pem-signature**、および**pkilocal.pem**はAC1からFTP経由でクライアントにダウンロードされます(詳細は省略します)。
3. ダウンロードした証明書ファイル(pkicachain.pem、pkilocal.pem-signature、およびpkilocal.pem:暗号化)をクライアントからFTP経由でAC2にアップロードします(詳細は省略します)。
4. 次の手順で、証明書ファイルをAC2にインポートします。

   #CRLチェックを無効にします。(必要に応じてCRLチェックを有効にできます。この例では、CRLチェックは不要であると想定しています。)

<AC2> system-view

[AC2]pki domain importdomain

[AC2-pki-domain-importdomain] undo crl check enable

#署名用のRSAキーペアを署名として指定し、暗号化用のRSAキーペアを暗号化として指定します。

[AC2-pki-domain-importdomain] public-key rsa signature name sign encryption name encr

[AC2-pki-domain-importdomain] quit

#PEM形式のCA証明書ファイルpkicachain.pemをPKIドメインにインポートします。

[AC2] pki import domain importdomain pem ca filename pkicachain.pem

#PEM形式のローカル証明書ファイルpkilocal.pem-signatureをPKIドメインにインポートします。証明書ファイルにはキーペアが含まれています。

[AC2] pki import domain importdomain pem local filename pkilocal.pem-signature

Please input the password:******

#PEM形式のローカル証明書ファイルpkilocal.pem暗号化をPKIドメインにインポートします。証明書ファイルにはキーペアが含まれています。

[AC2] pki import domain importdomain pem local filename pkilocal.pem-encryption

Please input the password:******

設定の確認

AC2にインポートされたローカル証明書情報を表示します。

[AC2] display pki certificate domain importdomain local

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

98:2c:79:ba:5e:8d:97:39:53:00

Signature Algorithm: sha256WithRSAEncryption

Issuer: C=CN, L=shangdi, ST=pukras, O=OpenCA Labs, OU=docm, CN=subca1 Validity

Not Before: May 26 05:56:49 2011 GMT

Not After : Nov 22 05:56:49 2012 GMT

Subject: C=CN, O=OpenCA Labs, OU=Users, CN=subsign 11 Subject

Public Key Info:

Public Key Algorithm: rsaEncryption Public-

Key: (1024 bit)

Modulus:

00:9f:6e:2f:f6:cb:3d:08:19:9a:4a:ac:b4:ac:63:

ce:8d:6a:4c:3a:30:19:3c:14:ff:a9:50:04:f5:00:

ee:a3:aa:03:cb:b3:49:c4:f8:ae:55:ee:43:93:69:

6c:bf:0d:8c:f4:4e:ca:69:e5:3f:37:5c:83:ea:83:

ad:16:b8:99:37:cb:86:10:6b:a0:4d:03:95:06:42:

ef:ef:0d:4e:53:08:0a:c9:29:dd:94:28:02:6e:e2:

9b:87:c1:38:2d:a4:90:a2:13:5f:a4:e3:24:d3:2c:

bf:98:db:a7:c2:36:e2:86:90:55:c7:8c:c5:ea:12: 01:31:69:bf:e3:91:71:ec:21

Exponent: 65537 (0x10001)

X509v3 extensions:

X509v3 Basic Constraints:

CA:FALSE

Netscape Cert Type: SSL

Client, S/MIME

X509v3 Key Usage:

Digital Signature, Non Repudiation X509v3

Extended Key Usage:

TLS Web Client Authentication, E-mail Protection, Microsoft Smartcardlogin

Netscape Comment:

User Certificate of OpenCA Labs X509v3

Subject Key Identifier:

AA:45:54:29:5A:50:2B:89:AB:06:E5:BD:0D:07:8C:D9:79:35:B1:F5

X509v3 Authority Key Identifier:

keyid:70:54:40:61:71:31:02:06:8C:62:11:0A:CC:A5:DB:0E:7E:74:DE:DD

X509v3 Subject Alternative Name: email:subsign@docm.com

X509v3 Issuer Alternative Name:

DNS:subca1@docm.com, DNS:, IP Address:1.1.2.2, IP Address:2.2.1.1 Authority Information Access:

CA Issuers - URI:http://titan/pki/pub/cacert/cacert.crt OCSP - URI:http://titan:2560/

1.3.6.1.5.5.7.48.12 - URI:http://titan:830/

X509v3 CRL Distribution Points: Full Name:

Signature Algorithm: sha256WithRSAEncryption 18:e7:39:9a:ad:84:64:7b:a3:85:62:49:e5:c9:12:56:a6:d2:
46:91:53:8e:84:ba:4a:0a:6f:28:b9:43:bc:e7:b0:ca:9e:d4:
1f:d2:6f:48:c4:b9:ba:c5:69:4d:90:f3:15:c4:4e:4b:1e:ef:
2b:1b:2d:cb:47:1e:60:a9:0f:81:dc:f2:65:6b:5f:7a:e2:36:
29:5d:d4:52:32:ef:87:50:7c:9f:30:4a:83:de:98:8b:6a:c9:
3e:9d:54:ee:61:a4:26:f3:9a:40:8f:a6:6b:2b:06:53:df:b6:
5f:67:5e:34:c8:c3:b5:9b:30:ee:01:b5:a9:51:f9:b1:29:37:
02:1a:05:02:e7:cc:1c:fe:73:d3:3e:fa:7e:91:63:da:1d:f1:
db:28:6b:6c:94:84:ad:fc:63:1b:ba:53:af:b3:5d:eb:08:b3:
5b:d7:22:3a:86:c3:97:ef:ac:25:eb:4a:60:f8:2b:a3:3b:da:
5d:6f:a5:cf:cb:5a:0b:c5:2b:45:b7:3e:6e:39:e9:d9:66:6d:
ef:d3:a0:f6:2a:2d:86:a3:01:c4:94:09:c0:99:ce:22:19:84:
2b:f0:db:3e:1e:18:fb:df:56:cb:6f:a2:56:35:0d:39:94:34:
6d:19:1d:46:d7:bf:1a:86:22:78:87:3e:67:fe:4b:ed:37:3d:
d6:0a:1c:0b

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

08:7c:67:01:5c:b3:5a:12:0f:2f
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=CN, L=shangdi, ST=pukras, O=OpenCA Labs, OU=docm, CN=subca1 Validity
Not Before: May 26 05:58:26 2011 GMT
Not After : Nov 22 05:58:26 2012 GMT
Subject: C=CN, O=OpenCA Labs, OU=Users, CN=subencr 11 Subject
Public Key Info:
Public Key Algorithm: rsaEncryption Public-
Key: (1024 bit)
Modulus:
00:db:26:13:d3:d1:a4:af:11:f3:6d:37:cf:d0:d4:
48:50:4e:0f:7d:54:76:ed:50:28:c6:71:d4:48:ae:
4d:e7:3d:23:78:70:63:18:33:f6:94:98:aa:fa:f6:
62:ed:8a:50:c6:fd:2e:f4:20:0c:14:f7:54:88:36:
2f:e6:e2:88:3f:c2:88:1d:bf:8d:9f:45:6c:5a:f5:
94:71:f3:10:e9:ec:81:00:28:60:a9:02:bb:35:8b:
bf:85:75:6f:24:ab:26:de:47:6c:ba:1d:ee:0d:35:
75:58:10:e5:e8:55:d1:43:ae:85:f8:ff:75:81:03:
8c:2e:00:d1:e9:a4:5b:18:39
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Basic Constraints:
CA:FALSE
Netscape Cert Type:
SSL Server
X509v3 Key Usage:
Key Encipherment, Data Encipherment
Netscape Comment:
VPN Server of OpenCA Labs X509v3
Subject Key Identifier:
CC:96:03:2F:FC:74:74:45:61:38:1F:48:C0:E8:AA:18:24:F0:2B:AB
X509v3 Authority Key Identifier:
keyid:70:54:40:61:71:31:02:06:8C:62:11:0A:CC:A5:DB:0E:7E:74:DE:DD

X509v3 Subject Alternative Name: email:subencr@docm.com
X509v3 Issuer Alternative Name:
DNS:subca1@docm.com, DNS:, IP Address:1.1.2.2, IP Address:2.2.1.1 Authority Information
Access:
CA Issuers - URI:http://titan/pki/pub/cacert/cacert.crt OCSP -
URI:http://titan:2560/
1.3.6.1.5.5.7.48.12 - URI:http://titan:830/

X509v3 CRL Distribution Points: Full Name:

Signature Algorithm: sha256WithRSAEncryption 53:69:66:5f:93:f0:2f:8c:54:24:8f:a2:f2:f1:29:fa:15:16:
90:71:e2:98:e3:5c:c6:e3:d4:5f:7a:f6:a9:4f:a2:7f:ca:af:
c4:c8:c7:2c:c0:51:0a:45:d4:56:e2:81:30:41:be:9f:67:a1:
23:a6:09:50:99:a1:40:5f:44:6f:be:ff:00:67:9d:64:98:fb:

72:77:9e:fd:f2:4c:3a:b2:43:d8:50:5c:48:08:e7:77:df:fb:
25:9f:4a:ea:de:37:1e:fb:bc:42:12:0a:98:11:f2:d9:5b:60:
bc:59:72:04:48:59:cc:50:39:a5:40:12:ff:9d:d0:69:3a:5e:
3a:09:5a:79:e0:54:67:a0:32:df:bf:72:a0:74:63:f9:05:6f:
5e:28:d2:e8:65:49:e6:c7:b5:48:7d:95:47:46:c1:61:5a:29:
90:65:45:4a:88:96:e4:88:bd:59:25:44:3f:61:c6:b1:08:5b:
86:d2:4f:61:4c:20:38:1c:f4:a1:0b:ea:65:87:7d:1c:22:be:
b6:17:17:8a:5a:0f:35:4c:b8:b3:73:03:03:63:b1:fc:c4:f5:
e9:6e:7c:11:e8:17:5a:fb:39:e7:33:93:5b:2b:54:72:57:72:
5e:78:d6:97:ef:b8:d8:6d:0c:05:28:ea:81:3a:06:a0:2e:c3:
79:05:cd:c3

CA証明書に関する詳細情報を表示するには、display pki certificate domainコマンドを使用します。

PKI設定のトラブルシューティング

ここでは、PKIに関する一般的な問題のトラブルシューティング情報について説明します。

CA証明書の取得に失敗しました

症状

CA証明書を取得できません。

解析

- ネットワーク接続がダウンしています。たとえば、ネットワークケーブルが破損しているか、コネクタの接触が不良です。
- 信頼できるCAが指定されていません。
- 証明書要求のURLが正しくないか、指定されていません。
- デバイスのシステム時間は、CAサーバーと同期されません。
- CAサーバーは、PKIドメインで指定された送信元IPアドレスを受け入れないか、送信元IPアドレスが指定されていません。
- ルートCA証明書のフィンガープリントが不正です。

ソリューション

1. ネットワーク接続の問題がある場合は、修正します。
2. PKIドメイン内の信頼できるCAおよび他のすべての必須パラメータを設定します。
3. pingコマンドを使用して、CAサーバーが到達可能であることを確認します。
4. デバイスのシステム時刻をCAサーバーと同期させます。
5. CAサーバーが受け入れることができる正しい送信元IPアドレスを指定します。正しい設定については、CA管理者に問い合わせてください。
6. CAサーバー上のCA証明書のフィンガープリントを確認します。
7. 問題が解決しない場合は、H3Cサポートに連絡してください。

ローカル証明書の取得に失敗しました

症状

ローカル証明書を取得できます。

解析

- ネットワーク接続がダウンしています。
- ローカル証明書要求を送信する前に、PKIドメインにはCA証明書がありません。
- LDAPサーバーが設定されていないか、正しく設定されていません。
- PKIドメイン内の証明書要求に対してキーペアが指定されていないか、指定されたキーペアがローカル証明書に含まれているキーペアと取得されたキーペアと一致しません。
- PKIドメインにPKIエンティティが設定されていないか、PKIエンティティの設定が正しくありません。
- CRLチェックはイネーブルですが、PKIドメインにはCRLがないため、CRLを取得できません。
- CAサーバーは、PKIドメインで指定された送信元IPアドレスを受け入れないか、送信元IPアドレスが指定されていません。

- デバイスのシステム時間は、CAサーバーと同期されません。

ソリューション

1. ネットワーク接続の問題がある場合は修正します。
2. CA証明書を取得またはインポートします。
3. 正しいLDAPサーバーのパラメータを設定します。
4. 証明書要求のキーペアを指定するか、既存のキーペアを削除し、新しいキーペアを指定して、ローカル証明書要求を再度送信します。
5. CAまたはRAの登録ポリシーをチェックし、PKIエンティティのアトリビュートがポリシー要件を満たしていることを確認します。
6. CRLリポジトリーからCRLを取得します。
7. CAサーバーが受け入れることができる正しい送信元IPアドレスを指定します。正しい設定については、CA管理者に問い合わせてください。
8. デバイスのシステム時刻をCAサーバーと同期させます。
9. 問題が解決しない場合は、H3Cサポートに連絡してください。

ローカル証明書の要求に失敗しました

症状

ローカル証明書要求は送信できません。

解析

- ネットワーク接続がダウンしています。たとえば、ネットワークケーブルが破損しているか、コネクタの接触が不良です。
- ローカル証明書要求が送信される前に、PKIドメインはCA証明書を持っていません。
- 証明書要求のURLが正しくないか、指定されていません。
- 証明書要求の受信権限が正しくないか、指定されていません。
- 必須のPKIエンティティパラメータが設定されていないか、正しく設定されていません。
- 証明書要求のPKIドメインにキーペアが指定されていないか、またはキーペアが証明書要求プロセス中に変更されています。
- 排他的な証明書要求アプリケーションは、PKIドメインで実行されています。
- CAサーバーは、PKIドメインで指定された送信元IPアドレスを受け入れないか、送信元IPアドレスが指定されていません。
- デバイスのシステム時間は、CAサーバーと同期されません。

ソリューション

1. ネットワーク接続の問題がある場合は、修正します。
2. CA証明書を取得またはインポートします。
3. pingコマンドを使用して、登録サーバーが到達可能であることを確認します。
4. certificate request fromコマンドを使用して、正しい証明書要求request fromコマンドを使用します。
5. CAまたはRA上の登録ポリシーで必要なPKIエンティティパラメータを設定します。
6. 証明書要求のキーペアを指定するか、既存のキーペアを削除し、新しいキーペアを指定して、ローカル証明書要求を再度送信します。
7. 証明書要求を中断するには、pki abort-certificate-request domainコマンドを使用します。
8. CAサーバーが受け入れることができる正しい送信元IPアドレスを指定します。正しい設定につい

ては、CA管理者に問い合わせてください。

9. デバイスのシステム時刻をCAサーバーと同期させます。
10. 問題が解決しない場合は、H3Cサポートに連絡してください。

CRLの取得に失敗しました

症状

CRLを取得できません。

解析

- ネットワーク接続がダウンしています。たとえば、ネットワークケーブルが破損しているか、コネクタの接触が不良です。
- PKIドメインは、CRLを取得しようとする前にCA証明書を持っていません。
- CRLリポジトリーのURLが設定されていないため、PKIドメイン内のCA証明書またはローカル証明書から取得できません。
- 指定されたCRLリポジトリーのURLが正しくありません。
- デバイスはSCEPを介してCRLを取得しようとしますが、次の問題が発生します。
  - PKIドメインはローカル証明書を持っていません。
  - 証明書内のキーペアが変更されました。
  - PKIドメインに、証明書要求用の不正なURLがあります。
- CRLリポジトリーは、CRLの配布にLDAPを使用します。ただし、LDAPサーバーのIPアドレスまたはホスト名は、CRLリポジトリーのURLには含まれず、PKIドメインにも構成されません。
- CAはCRLを発行しない。
- CAサーバーは、PKIドメインで指定された送信元IPアドレスを受け入れないか、送信元IPアドレスが指定されていません。

ソリューション

1. ネットワーク接続の問題がある場合は、修正します。
2. CA証明書を取得またはインポートします。
3. CRLリポジトリーのURLを取得できない場合は、次の条件が存在することを確認します。
   - 証明書要求のURLは有効です。
   - ローカル証明書が正常に取得されました。
   - ローカル証明書には、ローカルに保存されたキーペアと一致する公開鍵が含まれています。
4. LDAPサーバーアドレスがCRLリポジトリーURLに含まれているか、またはPKIドメイン内に構成されていることを確認してください。
5. CAサーバーがCRLの公開をサポートしていることを確認します。
6. CAサーバーが受け入れることができる正しい送信元IPアドレスを指定します。正しい設定については、CA管理者に問い合わせてください。
7. 問題が解決しない場合は、H3Cサポートに連絡してください。

CA証明書のインポートに失敗しました

症状

CA証明書をインポートできません。

解析

- CRLチェックはイネーブルですが、デバイスはPKIドメイン内にCRLを持たず、CRLを取得できません。
- CA証明書ファイルがインポートされる指定された形式は、実際の証明書ファイル形式と一致しません。

ソリューション

1. PKIドメイン内のCRLチェックをディセーブルにするには、undo crl check enableコマンドを使用します。
2. インポートしたファイルの形式が正しいことを確認します。
3. 問題が解決しない場合は、H3Cサポートに連絡してください。

ローカル証明書のインポートに失敗しました

症状

ローカル証明書をインポートできません。

解析

- PKIドメインにはCA証明書がなく、インポートされるローカル証明書ファイルにCA証明書チェーンが含まれていません。
- CRLチェックはイネーブルですが、デバイスはPKIドメイン内にCRLを持たず、CRLを取得できません。
- ローカル証明書ファイルがインポートされる指定された形式は、実際の証明書ファイル形式と一致しません。
- デバイスと証明書にはローカルキーペアがありません。
- 証明書が取り消されました。
- 証明書の有効期限が切れています。
- システム時刻が正しくありません。

ソリューション

1. CA証明書を取得またはインポートします。
2. undo crl check enableコマンドを使用して、CRLチェックをディセーブルにするか、証明書をインポートする前に正しいCRLを取得します。
3. インポートするファイルの形式が正しいことを確認します。
4. 証明書ファイルに秘密キーが含まれていることを確認します。
5. 証明書が取り消されていないことを確認します。
6. 証明書が有効であることを確認します。
7. デバイスの正しいシステム時刻を設定します。
8. 問題が解決しない場合は、H3Cサポートに連絡してください。

証明書のエクスポートに失敗しました

症状

証明書はエクスポートできません。

解析

- すべての証明書をPKCS12形式でエクスポートする場合、PKIドメインにはローカル証明書がありません。
- 指定されたエクスポートパスは存在しません。
- 指定されたエクスポートパスは無効です。
- エクスポートされるローカル証明書の公開鍵が、PKIドメインに設定されているキーペアの公開鍵と一致しません。
- デバイスのストレージスペースがいっぱいです。

ソリューション

1. 最初にローカル証明書を取得または要求します。
2. 必要なパスを作成するには、mkdirコマンドを使用します。
3. 正しいエクスポートパスを指定してください。
4. PKIドメイン内に正しいキーペアを設定します。
5. デバイスのストレージスペースをクリアします。
6. 問題が解決しない場合は、H3Cサポートに連絡してください。

ストレージパスの設定に失敗しました

症状

証明書またはCRLの格納パスを設定できません。

解析

- 指定されたストレージパスは存在しません。
- 指定されたストレージパスは無効です。
- デバイスのストレージスペースがいっぱいです。

ソリューション

1. パスを作成するには、mkdirコマンドを使用します。
2. 証明書またはCRLの有効な格納パスを指定します。
3. デバイスのストレージスペースをクリアします。
4. 問題が解決しない場合は、H3Cサポートに連絡してください。

公開鍵の管理

公開鍵の管理について

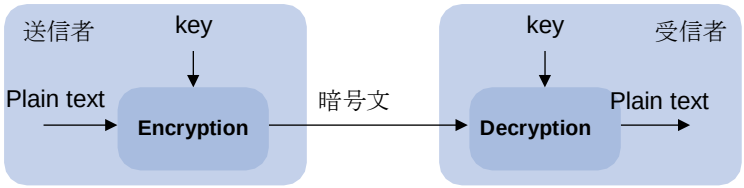
この章では、次の非対称キーアルゴリズムの公開鍵管理について説明します。

- Revest-Shamir-Adleman Algorithm(RSA)。
- デジタル署名アルゴリズム(DSA)。
- Elliptic Curve Digital Signature Algorithm(ECDSA)。

非対称キーアルゴリズムの概要

非対称キーアルゴリズムは、図1に示すように、セキュリティアプリケーションで2者間の通信を保護するために使用されます。非対称キーアルゴリズムでは、暗号化と復号化に2つの個別のキー(1つのパブリックキーと1つのプライベートキー)が使用されます。対称キーアルゴリズムでは、1つのキーのみが使用されます。

図1 暗号化と復号化



鍵の所有者はネットワーク上の平文で公開鍵を配布することができますが、秘密鍵はプライバシーに保持する必要があります。攻撃者がアルゴリズムと公開鍵を知っていても、秘密鍵を計算することは数学的に不可能です。

非対称鍵アルゴリズムの使用

セキュリティアプリケーション(SSH、SSL、およびPKIなど)は、次の目的で非対称キーアルゴリズムを使用します。

- **暗号化と復号化**: すべての公開鍵の受信者は、公開鍵を使用して情報を暗号化できますが、情報を復号化できるのは秘密キーの所有者だけです。
- **デジタル署名**: 鍵の所有者は秘密鍵を使用して、送信する情報にデジタル署名します。受信者は、送信者の公開鍵を使用して情報を復号化し、情報の信頼性を検証します。

RSA、DSA、およびECDSAはすべてデジタル署名を実行できますが、暗号化と復号化を実行できるのはRSAだけです。

公開鍵管理タスクの概要

公開鍵を管理するには、次のタスクを実行します。

1. ローカルキーペアの作成
2. ローカルホスト公開鍵の配布次のいずれかのタスクを選択します。
    - ホスト公開鍵のエクスポート

- ホスト公開鍵の表示

ピアデバイスがローカルデバイスを認証できるようにするには、ローカルデバイスの公開鍵をピアデバイスに配布する必要があります。

3. ピアホスト公開鍵の設定次のいずれかのタスクを選択します。
   - 公開鍵ファイルからのピアホスト公開鍵のインポート
   - ピアホスト公開鍵の入力

   ピアデバイスに送信される情報を暗号化するか、ピアデバイスのデジタル署名を認証するには、ローカルデバイス上でピアデバイスの公開鍵を設定する必要があります。
4. (任意)ローカルキーペアの破棄

ローカルキーペアの作成

制限事項およびガイドライン

ローカルキーペアを作成する場合は、次の注意事項に従ってください。

- キーアルゴリズムは、セキュリティアプリケーションで必要とされるものと同じである必要があります。
- RSAまたはDSAキーペアを作成する場合は、プロンプトで適切なキーモジュラス長を入力します。キーモジュラス長が長いほど、セキュリティが高くなり、キー生成時間も長くなります。

  ECDSAキーペアを作成する場合は、適切な楕円曲線を選択します。楕円曲線によってECDSAキー長が決定されます。キー長が長いほどセキュリティが高くなり、キー生成時間も長くなります。

  キー係数の長さおよびキーの長さの詳細については、表1を参照してください。
- キーペアに名前を割り当てない場合、デフォルト名がキーペアに割り当てられ、キーペアがデフォルトとしてマークされます。デフォルト名を別のキーペアに割り当てることもできますが、キーペアはデフォルトとしてマークされません。キーペアの名前は、同じキーアルゴリズムを使用する手動で名前を付けたすべてのキーペアの中で一意である必要があります。名前の競合が発生すると、既存のキーペアを上書きするかどうかを尋ねるメッセージが表示されます。
- キーペアは自動的に保存され、システムの再起動後も保持されます。

表1 異なるタイプの非対称鍵アルゴリズムの比較

タイプ	生成されたキーペア	係数/キーの長さ
RSA	• キーペア名を指定した場合は、1つのホストキーペア。 • キーペア名を指定しない場合は、1つのサーバキーペアと1つのホストキーペア。どちらのキーペアも既定の名前を使用します。 **注:** SSH 1.5だけがRSAサーバーのキーペアを使用します。	キー係数の長さ:512～2048ビット。デフォルト:1024ビット。 セキュリティを確保するには、最低768ビットを使用します。
DSA	1つのホストキーペア。	キー係数の長さ:512～2048ビット。デフォルト:1024ビット。 セキュリティを確保するには、最低768ビットを使用します。
ECDSA	1つのホストキーペア。	キーの長さ:192、256、384、または521ビット。

手順

1. システムビューに入ります。

   system-view
2. ローカルキーペアを作成します。

   public-key local create { dsa | ecdsa [secp192r1 | secp256r1 | secp384r1 | secp521r1] | rsa } [**name** *key-name*]

ローカルホスト公開鍵の配布

ローカルホストの公開鍵の配布について

ピアデバイスが次の操作を実行できるように、ピアデバイスにローカルホスト公開鍵を配布する必要があります。

- 公開鍵を使用して、ローカルデバイスに送信される情報を暗号化します。
- ローカルデバイスによって署名されたデジタル署名を認証します。

ローカルホストの公開鍵を配布するには、最初にキーをエクスポートまたは表示する必要があります。

- ホスト公開鍵をエクスポートします。
  - ホスト公開鍵をファイルにエクスポートします。
  - ホスト公開鍵をモニタ画面にエクスポートして、ファイルに保存します。

  キーをファイルにエクスポートした後、ファイルをピアデバイスに転送します。ピアデバイスで、ファイルからキーをインポートします。
- ホスト公開鍵を表示します。

  キーが表示されたら、キーを記録します。たとえば、フォーマットされていないファイルにコピーします。ピアデバイスでは、キーを文字どおり入力する必要があります。

ホスト公開鍵のエクスポート

制限事項およびガイドライン

ホスト公開鍵をエクスポートする場合は、次の制約事項およびガイドラインに従ってください。

- コマンドでファイル名を指定すると、指定したファイルにキーがエクスポートされます。
- ファイル名を指定しない場合は、キーがモニタ画面にエクスポートされます。エクスポートされたキーをファイルに手動で保存する必要があります。

手順

1. システムビューに入ります。

   system-view
2. ローカルホストの公開鍵をエクスポートします。
   - RSAホスト公開鍵をエクスポートします。

     public-key local export rsa [name *key-name* **] { openssh | ssh1 | ssh2 } [** *filename* **]**
   - ECDSAホスト公開鍵をエクスポートする。

     public-key local export ecdsa [name *key-name* **] { openssh | ssh2 } [** *filename* **]**
   - DSAホスト公開鍵をエクスポートします。

public-key local export dsa [name *key-name* **] { openssh | ssh2 } [** *filename* **]**

ホスト公開鍵の表示

任意のビューで次のタスクを実行します。

- ローカルRSA公開鍵を表示します。
  display public-key local rsa public [name *key-name* **]**
  RSAサーバーの公開鍵serverkey(デフォルト)をピアデバイスに配布しないでください。
- ローカルECDSA公開鍵を表示する。
  display public-key local ecdsa public [name *key-name* **]**
- ローカルDSA公開鍵を表示します。
  display public-key local dsa public [name *key-name* **]**

ピアホスト公開鍵の設定

ピアホスト公開鍵の設定について

ピアデバイスに送信される情報を暗号化するか、ピアデバイスのデジタル署名を認証するには、ローカルデバイス上でピアデバイスの公開鍵を設定する必要があります。

ピアホスト公開鍵を設定するには、次の方法を使用します。

- 公開鍵ファイルからピアホストの公開鍵をインポートします(推奨)。
- ピアホストの公開鍵を手動で入力(タイプまたはコピー)します。

デバイスのホスト公開鍵を取得する方法については、「ローカルホスト公開鍵の配布」を参照してください。

ピアホスト公開鍵設定に関する制約事項およびガイドライン

ピアホスト公開鍵を設定する場合は、次の制約事項および注意事項に従ってください。

- ピアホスト公開鍵を手動で入力する場合は、入力した鍵が正しい形式であることを確認してください。正しい形式でピアホスト公開鍵を取得するには、次の表示を使用します。
  public-key local publicコマンドを使用して、ピアデバイス上の公開鍵を表示し、キーを記録します。他の方法で表示される公開鍵のフォーマットは正しくない場合があります。キーが正しいフォーマットでない場合は、キーが廃棄され、エラーメッセージが表示されます。
- デバイスが記録されたピアホスト公開鍵の形式をサポートしているかどうかが不明な場合は、必ずピアホスト公開鍵を入力するのではなくインポートしてください。

公開鍵ファイルからのピアホスト公開鍵のインポート

このタスクについて

このタスクを実行する前に、ホスト公開鍵をピアデバイス上のファイルにエクスポートし、ピアデバイスからファイルを取得していることを確認してください。ホスト公開鍵のエクスポートの詳細は、「ホスト公開鍵のエクスポート」を参照してください。

キーをインポートすると、インポートされた公開鍵がPKCS(PKCS)形式の文字列に自動的に変換されます。

手順

1. システムビューに入ります。

system-view

2. 公開鍵ファイルからピアホスト公開鍵をインポートします。

   public-key peer *keyname* **import sshkey** *filename*

   デフォルトでは、ピアホストの公開鍵は存在しません。

ピアホスト公開鍵の入力

このタスクについて

このタスクを実行する前に、ピアデバイスにキーが表示され、キーが記録されていることを確認してください。ホスト公開鍵の表示の詳細は、「ホスト公開鍵の表示」を参照してください。

手順

1. システムビューに入ります。

   system-view

2. ピアホストの公開鍵の名前を指定し、公開鍵ビューを入力します。

   public-key peer *keyname*

3. キーを入力またはコピーします。

   スペースとキャリッジリターンを使用できますが、システムはそれらを保存しません。

4. 公開鍵表示を終了します。

   peer-public-key end

   公開鍵ビューを終了すると、ピアホストの公開鍵が自動的に保存されます。

ローカルキーペアの破棄

このタスクについて

セキュリティを確保するには、次のいずれかの状況でローカルキーペアを破棄し、新しいキーペアを生成します。

- ローカルキーが漏洩しました。侵入イベントが発生する可能性があります。
- デバイスのストレージメディアが交換されます。
- ローカル証明書の有効期限が切れています。ローカル証明書の詳細は、「PKIの構成」を参照してください。

手順

1. システムビューに入ります。

   system-view

2. ローカルキーペアを破棄します。

   public-key local destroy{ dsa | ecdsa | rsa } [name *key-name* **]**

公開鍵の表示および保守コマンド

任意のビューで表示コマンドを実行します。

タスク	コマンド

ローカル公開鍵を表示します。	**display public-key local { dsa | ecdsa | rsa } public [name** *key-name*]
ピアホストの公開鍵を表示します。	**display public-key peer [brief | name** *publickey-name*]

公開鍵管理の例

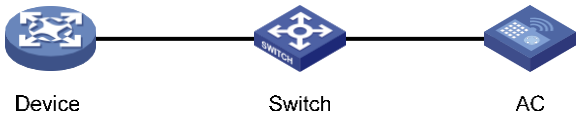
例:ピアホスト公開鍵の入力

ネットワーク構成

図2に示すように、不正アクセスを防止するために、ACはデジタル署名を使用してデバイスを認証します。ACで認証パラメータを設定する前に、次の手順を使用してAC上のデバイスの公開鍵を設定します。

- デバイス上にRSAキーペアを作成し、RSAホスト公開鍵をファイルにエクスポートします。
- AC上のデバイスのRSAホスト公開鍵を手動で指定します。

図2 ネットワーク図



手順

1. デバイスを次のように設定します。

   #デバイス上のデフォルト名を使用してローカルRSAキーペアを作成し、デフォルトのキー係数の長さ(1024ビット)を使用します。

   <Device> system-view

   [Device] public-key local create rsa

   The range of public key modulus is (512 ~ 2048).

   If the key modulus is greater than 512, it will take a few minutes. Press CTRL+C to abort.

   Input the modulus length [default = 1024]: Generating Keys...

   ....++++++

   ....++++++

   .....++++++++

   ....++++++++

   Create the key pair successfully.

   #すべてのローカルRSA公開鍵を表示します。

   [Device] display public-key local rsa public

   ==========

   Key name: hostkey (default)

   Key type: RSA

Time when key pair created: 16:48:31 2015/05/12

Key code:

30819F300D06092A864886F70D010101050003818D0030818902818100DA3B90F59237347B

8D41B58F8143512880139EC9111BFD31EB84B6B7C7A1470027AC8F04A827B30C2CAF79242E

45FDFF51A9C7E917DB818D54CB7AEF538AB261557524A7441D288EC54A5D31EFAE4F681257

6D7796490AF87A8C78F4A7E31F0793D8BA06FB95D54EBB9F94EB1F2D561BF66EA27DFD4788

CB47440AF6BB25ACA50203010001

==========

Key name: serverkey (default) Key

type: RSA

Time when key pair created: 16:48:31 2015/05/12 Key

code:

307C300D06092A864886F70D0101010500036B003068026100C9451A80F7F0A9BA1A90C7BC

1C02522D194A2B19F19A75D9EF02219068BD7FD90FCC2AF3634EEB9FA060478

DD0A1A49ACE

E1362A4371549ECD85BA04DEE4D6BB8BE53B6AED7F1401EE88733CA3C4CED3

91BAE633028A AC41C80A15953FB22AA30203010001

2. ACを次のように設定します。

#公開鍵表示でデバイスのホスト公開鍵を入力します。キーは、デバイスに表示されるものと文字どおり同じである必要があります。

<AC> system-view

[AC] public-key peer device

Enter public key view. Return to system view with "peer-public-key end" command.

[AC-pkey-public-key-

device]30819F300D06092A864886F70D010101050003818D003081890

2818100DA3B90F59237347B

[AC-pkey-public-key-device]

8D41B58F8143512880139EC9111BFD31EB84B6B7C7A1470027A

C8F04A827B30C2CAF79242E

[AC-public-key-device] 45FDFF51A9C7E917DB818D54CB7AEF538AB261557524A7441D2

88EC54A5D31EFAE4F681257

[AC-pkey-public-key-device]

6D7796490AF87A8C78F4A7E31F0793D8BA06FB95D54EBB9F94E

B1F2D561BF66EA27DFD4788

[AC-pkey-public-key-device] CB47440AF6BB25ACA50203010001

#公開鍵を保存して、システムビューに戻ります。

[AC-pkey-public-key-device] peer-public-key end

設定の確認

#ACに設定されているピアホストの公開鍵が、デバイスに表示されているキーと同じであることを確認します。

[AC] display public-key peer name device

==========

Key name: device Key

type: RSA

Key modulus: 1024 Key

code:

```
    30819F300D06092A864886F70D010101050003818D0030818902818100DA3B90F59237347B
    8D41B58F8143512880139EC9111BFD31EB84B6B7C7A1470027AC8F04A827B30C2CAF79242E
    45FDFF51A9C7E917DB818D54CB7AEF538AB261557524A7441D288EC54A5D31EFAE4F681257
    6D7796490AF87A8C78F4A7E31F0793D8BA06FB95D54EBB9F94EB1F2D561BF66EA27DFD4788
    CB47440AF6BB25ACA50203010001
```

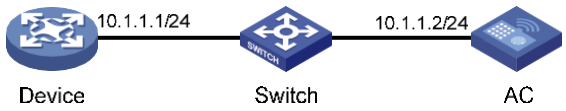
例:公開鍵ファイルからの公開鍵のインポート

ネットワーク構成

図3に示すように、ACはデジタル署名によってデバイスを認証します。ACで認証パラメータを設定する前に、次の手順を使用してAC上のデバイスの公開鍵を設定します。

- デバイス上にRSAキーペアを作成し、RSAホスト公開鍵をファイルにエクスポートします。
- デバイスのRSAホスト公開鍵を公開鍵ファイルからACにインポートします。

図3 ネットワーク図



手順

1. デバイスを次のように設定します。

   #デバイス上にデフォルト名を持つローカルRSAキーペアを作成し、デフォルトのモジュラス長(1024ビット)を使用します。

   <Device> system-view

   [Device] public-key local create rsa

   The range of public key modulus is (512 ~ 2048).

   If the key modulus is greater than 512, it will take a few minutes. Press CTRL+C to abort.

   Input the modulus length [default = 1024]:

   Generating Keys...

   ....++++++

   ....++++++

   .....++++++++

   ....++++++++

   Create the key pair successfully.

   #すべてのローカルRSA公開鍵を表示します。

   [Device] display public-key local rsa public

   ==========

   Key name: hostkey (default)

   Key type: RSA

Time when key pair created: 16:48:31 2015/05/12 Key

code:

30819F300D06092A864886F70D010101050003818D0030818902818100DA3B90F59237347B

8D41B58F8143512880139EC9111BFD31EB84B6B7C7A1470027AC8F04A827B30C2CAF79242E

45FDFF51A9C7E917DB818D54CB7AEF538AB261557524A7441D288EC54A5D31EFAE4F681257

6D7796490AF87A8C78F4A7E31F0793D8BA06FB95D54EBB9F94EB1F2D561BF66EA27DFD4788

CB47440AF6BB25ACA50203010001

==========

Key name: serverkey (default) Key

type: RSA

Time when key pair created: 16:48:31 2015/05/12 Key

code:

307C300D06092A864886F70D0101010500036B003068026100C9451A80F7F0A9BA1A90C7BC

1C02522D194A2B19F19A75D9EF02219068BD7FD90FCC2AF3634EEB9FA060478

DD0A1A49ACE

E1362A4371549ECD85BA04DEE4D6BB8BE53B6AED7F1401EE88733CA3C4CED3

91BAE633028A AC41C80A15953FB22AA30203010001

#RSAホスト公開鍵をファイルdevice.pub. [Device] public-key

local export rsa ssh2 device.pub [Device] quit

#FTPサーバー機能をイネーブルにし、ユーザー名ftpおよびパスワード123を使用してFTPユーザーを作成し、FTPユーザロールをnetwork-adminに設定します。

[Device] ftp server enable

[Device] local-user ftp

[Device-luser-manage-ftp] password simple 123

[Device-luser-manage-ftp] service-type ftp

[Device-luser-manage-ftp] authorization-attribute user-role network-admin

[Device-luser-manage-ftp] quit

2. ACを次のように設定します。

#デバイスから公開鍵ファイルdevice.pubを取得するには、バイナリモードでFTPを使用します。

<AC> ftp 10.1.1.1

Connected to 10.1.1.1 (10.1.1.1).

220 FTP service ready.

User (10.1.1.1:(none)): ftp

331 Password required for ftp.

Password:

230 User logged in.
Remote system type is UNIX.
Using binary mode to transfer files.

ftp> binary
200 TYPE is now 8-bit binary
ftp> get device.pub
227 Entering Passive Mode (10,1,1,1,118,252)
150 Accepted data connection
226 File successfully transferred
301 bytes received in 0.003 seconds (98.0 kbyte/s)

ftp> quit
221-Goodbye. You uploaded 0 and downloaded 1 kbytes.
221 Logout.
#キーファイルdevice.pubからホストの公開鍵をインポートします。
<AC> system-view
[AC] public-key peer device import sshkey device.pub

設定の確認

#ACに設定されているピアホストの公開鍵が、デバイスに表示されているキーと同じであることを確認します。
[AC] display public-key peer name device

==========
Key name: device
Key type: RSA
Key modulus: 1024
Key code:
30819F300D06092A864886F70D010101050003818D0030818902818100DA3B90F59237347B
8D41B58F8143512880139EC9111BFD31EB84B6B7C7A1470027AC8F04A827B30C2CAF79242E
45FDFF51A9C7E917DB818D54CB7AEF538AB261557524A7441D288EC54A5D31EFAE4F681257
6D7796490AF87A8C78F4A7E31F0793D8BA06FB95D54EBB9F94EB1F2D561BF66EA27DFD4788
CB47440AF6BB25ACA50203010001